PHYSICAL DESIGN FOR PERFORMANCE AND THERMAL AND POWER-SUPPLY RELIABILITY IN MODERN 2D AND 3D MICROARCHITECTURES

A Dissertation
Presented to
The Academic Faculty

By

Michael B. Healy

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy
in
Electrical and Computer Engineering

School of Electrical and Computer Engineering
Georgia Institute of Technology
December 2010

PHYSICAL DESIGN FOR PERFORMANCE AND THERMAL AND POWER-SUPPLY RELIABILITY IN MODERN 2D AND 3D MICROARCHITECTURES

Approved by:

Dr. Hsien-Hsin S. Lee, Committee Chair
Assoc. Professor, School of ECE
Georgia Institute of Technology

Dr. Saibal Mukhopadhyay
Asst. Professor, School of ECE
Georgia Institute of Technology

Dr. Sung Kyu Lim, Advisor
Assoc. Professor, School of ECE
Georgia Institute of Technology

Dr. Hyesoon Kim
Asst. Professor, College of Computing
Georgia Institute of Technology

Dr. Gabriel H. Loh
Assoc. Professor, College of Computing
Georgia Institute of Technology

Date Approved: August 27, 2010

Dedicated to my parents, Peter and Laurie Healy, for their boundless love and support.

ACKNOWLEDGMENT

I would like to express my deepest gratitude to Professor Sung Kyu Lim for his guidance of my research during my Ph.D. study at Georgia Tech. I would also like to thank my defense committee members, Professor Hsien-Hsin Sean Lee, Professor Gabriel H. Loh, Professor Saibal Mukhopadhyay, and Professor Hyesoon Kim, for their valuable suggestions. I am especially thankful to my reading committee members, Professor Lee and Professor Loh, for their extensive help throughout my research career at Georgia Tech. I also would like to thank Professor Linda Milor for serving on my thesis proposal committee. I would like to thank all the members of the CREST, GTCAD, and MARS groups for their support and friendship, especially Mongkol Ekpanyapong, Jacob Minz, Ismail F. Baskaya, Pinar Korkmaz, Chinnakrishnan S. Ballapuram, Kiran Puttaswamy, Mario Vittes, Fayez Mohamood, Sasank Reddy, Somaskanda Thyagaraja, Wook-Jin Chung, Brian Smith, Eric Wong, Mohit Pathak, Dae Hyun Kim, Xin Zhao, Young-Joon Lee, Dean L. Lewis, Krit Athikulwongse, Moongon Jung, Chang Liu, Hemant Sane, Brian Oullette, Rohan Goel, and Thomas Moon, who have collectively helped me in countless ways throughout my stay at Georgia Tech. I would like to especially thank Kathlyn Carroll for her help in editing this dissertation, and Dae Hyun Kim for his friendship and the extensive discussions we shared regarding each other's research.

My family has also been extremely supportive of me throughout my life, and I wish to express my deepest gratitude and love to them for everything they have provided me. Lastly, I would like to thank the organizers, faculty, staff, and students of the Texas Academy of Mathematics and Science for providing me with life-changing opportunities and support; I would not be the person I am today without the experiences I gained there.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

The main objective of this research is to examine the performance, power noise, and thermal trade-offs in modern traditional (2D) and three-dimensionally-integrated (3D) architectures and to present design automation tools and physical design methodologies that enable higher reliability while maintaining microarchitectural performance for these systems. Five main research topics that support this goal are included. The first topic focuses on thermal reliability. The second, third, and fourth, topics examine power-supply noise. The final topic presents a set of physical design and analysis methodologies used to produce a 3D design that was sent for fabrication in March of 2010.

The first section of this dissertation details a microarchitectural floorplanning algorithm that enables the user to choose and adjust the trade-off between microarchitectural performance and general operating temperature in both 2D and 3D systems, which is a major determinant of overall reliability and chip lifetime. Simulation results demonstrate that the algorithm performs as expected and successfully provides the user with the desired trade-off. The first section also presents a thermal-aware microarchitectural floorplanning algorithm designed to help reduce the operating temperature of the cores in the unique environment present within multi-core processors. Heat-coupling between neighboring cores is considered during the optimization process to provide floorplans that result in lower maximum temperature.

The second section explores power-supply noise in processors caused by fine-grained clock-gating and describes a floorplanning algorithm created to work with an active noise-canceling clock-gating controller. Simulation results show that combining these two techniques results in lower power-supply noise with minimal processor performance impact. The third section turns to future 3D systems with a large number of stacked active layers (many-tier systems) and examines power-supply delivery challenges in these systems. Parasitic resistance, capacitance, and inductance are calculated for the 3D vias, and the results

of scaling various parameters in the power-supply-network design are presented. Several techniques for reducing power-supply-network noise in these many-tier systems are explored. The fourth section describes a layout-level analysis of a novel power distribution through-silicon-via topology and it's effect on IR-drop and dynamic noise. Simulations show that both types of power-supply noise can be reduced by more than 20% in systems with non-uniform per-tier power dissipation when using the proposed topology.

The final section explains the physical design and analysis techniques used to produce the layouts for 3D-MAPS, a 64-core 3D-stacked memory-on-processor system targeted at demonstration of large memory bandwidth using 3D connections. The 3D-aware physical design flow utilizing non-3D-aware commercial tools is detailed, along with the techniques and add-ons that were developed to enable this process.

CHAPTER 1

INTRODUCTION AND BACKGROUND

1.1 Introduction

Reliability of microprocessors has become of increasing concern to designers in recent years. Shrinking transistor feature sizes have increased the rate at which the performance of devices degrades over time. Concurrently, advancing core-level microprocessor performance has reached the so-called ”power wall.” At this point, increases in per-core performance, mainly driven by increases in operating frequency, lead to increases in power consumption faster than heat can be economically removed from the package. Furthermore, supply-voltage scaling has increased the magnitude of the current demanded by high-performance microprocessors, while at the same time forcing tighter restrictions on power-supply variations. These tighter restrictions create ever-increasing pressure on the overheads required to create reliable power-supply designs.

Additionally, three-dimensionally integrated circuit (3D IC) technology has recently come to the forefront of the semiconductor industry’s interest. 3D integration is the process of stacking multiple active layers of silicon together with a high-density interconnect between them. 3D ICs offer a potential way to decrease power consumption and delay by reducing off-chip communication and wirelength. However, 3D ICs also suffer from increased thermal problems because of the larger thermal resistance between the active circuitry and the heatsink. Physical design tools that are 3D-aware are not yet readily available, and are a topic of active research. Best practices and design methods for 3D ICs are also under active development.

These and other factors create the need for reliability-aware and 3D-aware automated physical design tools useful for early-stage designers as well as general physical design

techniques and methods that improve integrated circuit reliability for 3D circuits and microprocessors. This dissertation examines performance, power noise, and thermal trade-offs in modern traditional (2D) and three-dimensionally integrated (3D) architectures and presents design automation tools and physical design methodologies that enable higher reliability while maintaining microarchitectural performance for these systems. Five main research topics that support this goal are included. The first section focuses on thermal reliability. The second, third, and fourth sections examine power-supply noise. The final section presents a set of physical design and analysis methodologies used to produce a 3D design that was sent for fabrication in March of 2010.

1.1.1 Contributions

This dissertation presents a set of design automation tools and design techniques that enhance thermal and power-supply reliability in modern microarchitectures and ICs. The main contributions of this research are:

- The first thermal-aware microarchitectural floorplanner for both 2D and 3D ICs. The thermal analyzer used during optimization includes dynamic power, leakage power, global interconnect power, and clock power for increased modeling accuracy. The microarchitectural floorplanner considers the interdependence of temperature and leakage current to effectively avoid thermal runaway.

- The first thermal-aware floorplanner to extend traditional floorplanning techniques to effectively optimize operating temperatures in multi-core processors with minimal performance impact. A number of multi-granularity temperature-optimizing floorplanning techniques are evaluated using an architecture with multiple cores and level-2 cache banks. The optimization accounts for the routing of the memory bus required to connect the cache banks and cores.

- A novel microarchitectural-level supply-noise controller and a floorplanning algorithm that works in concert with it to eliminate transient supply-noise violations.

Two new metrics are used to guide the floorplanner to optimize supply noise. Consideration of the physical basis that the microarchitectural controller uses to counter rapid current-demand changes leads to the complete elimination of supply-noise violations.

- An examination of thermal and power-supply noise scaling in large-scale 3D systems. A scalable prototype consisting of one layer containing a quad-core processor and eight layers containing DRAM is used to investigate the most important factors leading to power-supply noise in systems with up to 46 stacked layers. Results indicate that thermal issues can be controlled with micro-fluidic heatsinks and that power-supply-bump pitch will be increasingly important in large-scale systems.

- A novel through-silicon-via (TSV) topology for 3D power-supply networks that reduces both IR-drop and dynamic noise compared to a traditional topology. Layout-level analysis of a 3D-stacked memory-on-processor system shows that the novel TSV topology can reduce IR-drop by 21%, and dynamic noise by 32%, over a non-3D system, even though the 3D system consumes more power.

- The detailed methodology used to construct and analyze the layouts of 3D-MAPS, arguably the first many-core 3D processor in academia. The tool-flow used to design and analyze 3D-MAPS is based on commercial tools from Cadence, Synopsys, and Mentor Graphics that are not 3D-aware, along with the customizations needed to handle TSVs and 3D stacking.

1.1.2 Organization

This dissertation is organized into seven chapters as follows:

- **Chapter 1**: Introduces the thesis of this dissertation and provides motivation and background.

- **Chapter 2**: Presents a multi-objective microarchitectural floorplanner for 2D and 3D ICs and details a thermal-aware microarchitectural floorplanner for multi-core processors.

- **Chapter 3**: Introduces a microarchitectural-level dynamic power-supply-noise controller and a floorplanner designed specifically to work with it.

- **Chapter 4**: Provides a thermal and power-supply-noise scaling study of large-scale 3D systems.

- **Chapter 5**: Presents a layout-level analysis of a novel power-supply TSV topology and it's effect on IR-drop and dynamic noise.

- **Chapter 6**: Describes 3D-MAPS, a 64-core memory-on-processor 3D stacked system sent for fabrication in May of 2010, and the 3D-aware tool-flow used to produce its layouts.

- **Chapter 7**: Summarizes the research presented in this dissertation and provides concluding remarks.

1.2 Background

Reliability in integrated circuits (ICs) refers to the continuous correct operation of the device over its intended lifetime. Thermal reliability is related to both device aging and proper circuit operation. Power-supply reliability mainly impacts proper circuit operation. Temperature fluctuations in ICs cause many types of reliability problems. Mismatch between the coefficients of thermal expansion (CTE) for the various materials used to manufacture and package ICs causes mechanical stress during thermal cycling that can lead to delamination and solder fracturing. High operating temperatures are also a major cause of temporal device slowdown [2, 3]. Fluctations in power-supply voltage affect device switching speed, leakage current, and aging [2, 3]. Many techniques have been proposed at all stages of

the design process to deal with reliability problems in ICs. Since reliability problems are caused by high operating temperatures and high power-supply-noise levels, techniques or methods that reduce operating temperatures or power-supply-noise levels will improve reliability, which is a major underpinning of this dissertation. This chapter summarizes prior work that is related to the design tools and techniques presented by this dissertation.

The very-large-scale integration (VLSI) design process consists of system architecture description, circuit design, logic synthesis, and physical design. Physical design is the process of determining the exact co-ordinates of all of the circuit components implementing a given logical circuit description. Physical design obviously has a large impact on the two main reliability issues that are the focus of this dissertation: namely, power-supply reliability and thermal reliability. Accordingly, the work discussed in the following chapters deals with physical design, and presents both automated tools as well as design techniques that improve these types of reliability.

The main steps of physical design are partitioning, floorplanning, power-supply-network routing, gate placement, clock-tree synthesis, and signal routing. Floorplanning is an early-stage planning process that is performed to determine the relative location of large groups of gates within a design. The floorplanning process can have a large impact on the final design's wirelength, critical-path delay, power distribution, and thermal distribution. Microarchitectural floorplanning is a specific variant of the floorplanning problem focused on determining the location and aspect ratio of the abstract modules in a microprocessor. The following section describes previous work on floorplanning. The subsequent sections describe prior research targeted at enhancing either thermal or power-supply reliability during various stages of physical design or microarchitecture design. The final sections focus on 3D IC design and the challenges faced by 3D IC designers.

1.2.1 Floorplanning

1.2.1.1 Traditional Floorplanning

Floorplanning is the process of determing the aspect ratio and location of a set of modules representing large collections of gates. These modules are typically presented with a given area and aspect ratio bounds. Floorplanning algorithms can be classified into two categories: slicing floorplanners and non-slicing floorplanners. Slicing floorplanners [4, 5, 6] generate slicing structures, which are rectangular bisections that can be optained by recursively subdividing rectangles either horizontally or vertically. The floorplanning solution space is not completely covered by slicing floorplans, though in practice it is generally possible to construct slicing floorplans with performance near that of non-slicing floorplans. Nonetheless, non-slicing floorplanning algorithms [7, 8, 9, 10] have become dominant in both the research and industrial communities. The most commonly used non-slicing algorithms are based on mathematical programming [7] or heuristics [8].

1.2.1.2 Microarchitectural Floorplanning

Microarchitectural floorplanning was first examined as a distinct subproblem of traditional floorplannning in 2003 by Cong *et al.* [11]. The authors proposed a simultaneous microarchitecture and physical design optimization engine called MEVA. MEVA has three main components: a physical design engine, a timing estimator, and a cycle-level microarchitectural simulator. MEVA requires a set of architectural alternatives, such as cache sizes and number of functional units, and the physical design information associated with those alternatives as inputs. The architectural alternatives required by MEVA represent the physical constraints and timing information of architectural modules, such as the size and timing of multiple different cache sizes. MEVA's physical design engine is based on simulated annealing optimization and uses a cost function that includes timing-slack-weighted wirelength and instructions per cycle (IPC). MEVA also uses a novel annealing move that changes the configuration of the microarchitecture being evaluated. By combining the

use of a timing estimator at every annealing move and the cycle-level simulator's estimation of IPC, MEVA is able to simultaneously optimize cycle time and IPC. This work is highly limited by its combined use of simulated annealing, which requires examination of a large number of candidate outputs, and cycle-level architectural simulation, which is extremely time-consuming. Together these factors combine to create very large run-time for the MEVA algorithm. Adding reliability optimization to the MEVA algorithm would extend its run-time to unacceptable levels.

Nearly concurrently with Cong *et al.* [11], Casu *et al.* [12] proposed a method for considering throughput during floorplanning optimization. Their method does not target microarchitectures, but the technique is well suited for microarchitectural floorplanning. Casu *et al.* began with a formal definition of throughput in the case of latched interconnects. Directed loops in latched interconnect systems can impact the correctness of the system. To ensure the correctness of the system a delay must be cyclically added. Casu *et al.* built a detailed definition of throughput and then observe that this definition does not lend itself well to the problem of annealing-based floorplan optimization because it violates two requirements. First, formulating a cost function from the throughput is computationally intensive. Second, such a cost function will have strong discontinuities. Casu *et al.* then went on to formulate a function that is quickly calculated, smooth, and computed over the whole netlist. When combined with annealing, this cost function allows Casu *et al.* to optimize a weighted combination of area, wirelength, and throughput. However, this work does not consider system reliability.

In the next year, Ekpanyapong *et al.* [13] formulated a new way to optimize architectural performance through physical design. The authors make the observation that, as a result of shrinking transistor sizes, wire delay will dominate gate delay in the near future. This increase in the wire delay to gate delay ratio is driving an increase in the number of flip-flops added to global interconnects. Next, Ekpanyapong *et al.* formulated a new metric called profile weight. The profile weight is the normalized communication frequency of

a net. When a net is used more frequently, its profile weight will be high. When a net is not used very often, it will have a low profile weight. Ekpanyapong *et al.* showed that by selectively optimizing the length of wires with high profile weight, the overall performance of a microarchitecture (measured in IPC) can be increased compared to optimizing pure wirelength. The authors accomplish this using a modified microarchitectural simulator that takes into account a cycle-time-based calculation of the pipeline delay along the important global interconnects. Ekpanyapong *et al.*'s optimization engine is a novel method utilizing a linear-programming-based (LP) partitioner, which determines relative location, and then an LP-based floorplanner. The LP-based floorplanner uses the relative locations determined during the partitioning steps to remove integer variables from an LP formulation of the floorplanning problem. This work does not consider the operating temperature of the resulting design as an objective, and the authors do not report thermal results.

Simultaneously with Ekpanyapong *et al.* [13], Long *et al.* [14] proposed a method for architectural performance optimization using a piecewise-linear estimation model for the architecturally-important global interconnects. Long *et al.* simulated the cycles per instruction (CPI) of their chosen architecture while varying the length (i.e. the number of pipeline stages) of each interconnect. They then build a piecewise-linear model of each interconnect's effect on CPI. During optimization, the linear combination of all the various interconnects' impacts on CPI is used to calculate the CPI portion of the cost function. Long *et al.* also used simulated-annealing-based optimization with a cost function that considers area, wirelength, and CPI. Again, this work does not consider a thermal objective.

Finally, Nookala *et al.* [15] proposed a method similar to that of Long *et al.* Nookala *et al.* formulated an approximation of performance based on the mathematical concept of the statistical design of experiments [16]. The authors establish a list of the potentially significant interactions between the delays of the various interconnects and find a design exploration heuristic that provides the minimum number of experiments to ensure accuracy.

Their chosen heuristic requires only 32 experiments (simulations) to build a table of the interacting factors. This table is then used to estimate the performance of the architecture with any combination of interconnect lengths. Nookala *et al.* demonstrated that this technique allows them to effectively optimize the performance of the architectural floorplan. This work also does not consider a thermal objective.

In summary, initial work on microarchitectural floorplanning used full sets of cycle-level simulations and timing analyzers to estimate performance during optimization [11]. Subsequent works have focused on making simplifications of those performance models to speed up optimization: first, with profile-based net weighting [13] and piecewise-linear interconnect performance models [14], and finally, with a statistical design of experiments [15]. The work presented in this dissertation is the first to extend prior art to simultaneously consider thermal and performance objectives for both 2D and 3D floorplans.

1.2.2 Thermal Reliability

IC operating temperature is of significant concern to designers because it is a large factor in many causes of reduced device reliability. For example, co-efficient of thermal expansion (CTE) mismatches between silicon layers and organic packaging substrates can cause delamination, cracking, and mechanical failure of C4 bumps and packages [17] when the operating temperature of the IC is high. The operating temperature of ICs is influenced mainly by joule heating caused by current flowing through the device's resistive materials. The majority of this heating occurs within the transistor channels. The detailed thermal profile of a design is therefore a function of the spatial distribution of transistors and their temporal activity. The thermal time constant of a packaged IC ($\approx 100ms$) is much larger than the typical cycle time of those ICs ($< 10ns$) [18], so the average power behavior of floorplan-level modules is a reasonable input for thermal simulations. The following subsections discuss some prior art for physical and microarchitectural design for thermal reliability.

1.2.2.1 Physical Design for Thermal Reliability

Previous work on physical design for thermal reliability has focused on thermal-aware placement [19, 20, 21, 22, 23] that targets only circuit-level designs. Placement algorithms are targeted at gate-level cells and netlists and are inefficient for handling the large multi-pin nets and soft-module constraints of the floorplanning problem. There are few works targeting thermal-aware floorplanning. For example, Cong *et al.* [24] presented a 3D temperature-driven floorplanner based on the transitive closure-graph [25] floorplan representation and a novel bucket structure to represent module overlap. They use various thermal analyzers to trade off execution time with accuracy and overall performance, and demonstrate that floorplanning can have a significat impact on device operating temperature. However, they do not consider any other non-standard performance metrics, such as architectural performance, and only report results for generic benchmark circuits. Hung *et al.* [26] presented a thermal-aware floorplanner based on genetic algorithms, but their solution does not provide faster execution time or better results than previously published works and also considers only thermal, area, and wire-length optimization objectives. Ekpanyapong [27] demonstrated that taking advantage of architectural information during the physical design stage can lead to significant improvement in performance. Therefore, physical design techniques that are not architecture-aware are not appropriate for use in designing high-performance microprocessors. This dissertation presents the first works that focus on thermal-aware floorplanning for 2D, 3D, and multi-core microprocessors that optimizes microarchitectural performance, operating temperature, and traditional physical design objectives.

1.2.2.2 Microarchitecture Design for Thermal Reliability

Several microarchitecture research works on temperature [28, 29, 30] and leakage power [31, 32, 33, 34, 35] reduction provide run-time management of the functional modules but do not perform floorplanning. He *et al.*'s work [35], the most recently published, presented a system-level leakage power model and discusses dynamic management to reduce the

thermal problem. Their work also discusses thermal runaway and shows that a dynamic management scheme must include consideration of leakage power to be effective.

In the area of multi-core microarchitectural work Li *et al.* [36] analyzed a networked memory subsystem by considering thermal effects and examined the impact of exchanging bank locations with core locations in a 3D chip-stack. Chaparro *et al.* [37] studied the thermal implications of a 16-core architecture and explored various thermal management schemes, such as activity migration. Ogras *et al.* [38] proposed a methodology for customized communication-architecture synthesis that minimizes total energy consumption while satisfying many communication pattern requirements. It should be noted that the microarchitectural design techniques for thermal reliability described in this subsection can be used in concert with physical design techniques presented in this dissertation.

1.2.3 Power-Supply Noise

Power-supply networks are designed to supply a given voltage difference between the power and ground supply lines for the gates in a design. The actual voltages seen by the gates will vary from the nominal value because of several factors. These include both resistive losses and interactions between the capacitive and inductive parasitics of the power distribution network. Deviations in the voltage level cause reductions in device speed, accelerate device aging, and can cause functional failures. A supply-voltage variation tolerance of 10% of the nominal supply voltage is generally used to ensure reliable circuit operation [39].

There are two major types of non-ideal behavior in power-supply networks. The first is called "IR-drop," and results from dynamic power dissipated by switching transistors. This dynamic power dissipation creates current flow through the power-supply network. The current flow induces resistive voltage drops according to Ohm's Law, $V = I \times R$, where V is the voltage drop, I is the current, and R is the resistance of the power-supply network. The second type of non-ideal power-supply behavior is called "transient noise," "dynamic noise," or "LdI/dt" noise, and is a result of the time-varying nature of the current demand in

active circuits. Time-varying current demands interact with the parasitic inductances and capacitances in power-distribution networks to produce oscillatory voltage swings. The magnitude of the voltage swings is a function of the package-level inductance, the on-die decoupling capacitance (decap), and the magnitude and timing of the current demands.

The largest factor creating transient noise is the so-called "first droop" noise [40, 41] that results from the interaction of the inductance in the package and the on-chip decap during sleep transitions. Power gating is a method for reducing leakage power consumption in inactive portions of a circuit. In power gating, large transistors are added between the actual power supply and a "virtual supply" that is then distributed to the rest of the circuit. Power gating caused by sleep transitions creates large transient changes in the current demand on the power-supply network. In the case of 3D systems the TSVs add inductance of their own to that which naturally exists in the package, which increases the power-supply noise in these systems.

The second largest factor creating transient noise is high-speed clock gating [42]. Clock gating is a technique designed to save dynamic power dissipation by preventing unused portions of a circuit or pipeline stage from switching. This is accomplished by stopping the edges of the clock signal before they reach the unused circuitry. Aggressive clock gating causes large cycle-by-cycle changes in the current demand of ICs and therefore is another source of LdI/dt noise. The following subsections present a summary of prior work relating to physical-design algorithms and microarchitectural design techniques for increasing power-supply-network reliability.

1.2.3.1 Physical Design for Power-Supply-Network Reliability

Power-supply-noise-aware floorplanning has previously been studied [43, 44, 45, 46] by the design automation community. The central idea of these works involves two concepts: the first idea is to create a low impedance path to the power supply, and the second is to optimize on-chip decap placement and allocation to suppress inductive noise effects. Zhao *et al.* [43] presented both a decap placement algorithm and a noise-aware

floorplanning algorithm. Their decap placement algorithm inserts decaps in the whitespace of the floorplan and expands the available whitespace until the power-supply noise is below a threshold. Their noise-aware floorplanning algorithm is based on simulated annealing with the Sequence-Pair floorplan representation and uses a linear program during the low-temperature moves to add a cost-function penalty related to the area and wirelength increases caused by decap insertion. This work is limited by its use of linear programming during the annealing process. This significantly increases the run-time of their technique. Chen *et al.* [44] presented an IR-drop-aware floorplanner based on simulated-annealing with the normalized Polish expression floorplan representation [5]. The cost function for their floorplanner includes a power-supply-noise factor calculated using a modified network-flow algorithm. This work is limited by its focus on steady-state IR-drop and cannot handle dynamic inductive noise events. Chen *et al.* [45] presented a priority assignment optimization and dynamic functional unit selection scheme that balances current demand in a floorplan-aware fashion. These techniques dynamically adjust the clock gating state and functional-unit instruction mapping in a floorplan-aware fashion. This work has limited application to general architectures because of its focus on wide-issue machines. Minz *et al.* [46] presented a simulated-annealing-based floorplanner and decap allocation flow for 3D System-on-Package (SOP). Their decap allocation algorithm uses a linear program to minimize the effective distance between decaps and floorplan modules while considering 3D connections. This work is also limited in application to reducing static IR-drop, and is incapable of optimizing inductive noise. In contrast to prior efforts, the work in this disseration is the first to present the combination of a microarchitectural-level high-frequency dynamic-supply-noise controller and an accompanying noise-aware floorplanning algorithm specifically designed to work with it.

1.2.3.2 Microarchitectural Design for Power-Supply-Network Reliability

The microarchitecture community has paid notable attention to LdI/dt issues caused by clock gating, and have proposed solutions to characterize and address them. Grochowski

et al. [42] was one of the first to illustrate the criticality of LdI/dt and propose solutions from the perspective of architects. Their work showed that applications exhibit varying current characteristics in a large range and proposed a microarchitectural solution to improve current demands with a feedback control mechanism. Their proposed mechanism can dynamically estimate supply-noise violations by performing current-to-voltage calculations on the chip. The mechanism then throttles instruction fetch or issue upon the detection of reliability emergencies. The work of Grochowski *et al.*, however, addressed the mid-frequency LdI/dt problem at the chip-level and the architecture presented is incapable of altering current demands over a smaller interval (e.g. less than 25 clock cycles). In contrast, the work presented in this dissertaion is targeted at mitigating high-frequency LdI/dt problems.

Joseph *et al.* [47, 48] analyzed power-supply response and control voltage emergencies in a processor via microarchitectural techniques. They concentrated on the worst case LdI/dt occuring at the *resonant frequency* of the power supply and proposed a solution to mitigate the mid-frequency LdI/dt problem in the range of 50-100 MHz. Similarly, Powell and Vijaykumar proposed techniques in [49] to mitigate reliability issues caused by the resonant frequency of the chip. Their technique exploits the resonant behavior of the power supply and dynamically alters the processor current frequency into the non-resonant range to avoid large voltage drops and spikes. Another technique called *Pipeline Damping* [50] by the same authors throttles microarchitectural activity at the front-end and the back-end to alter current surges at the resonant frequency.

In contrast to pure hardware-based techniques, Hazelwood and Brooks [51] proposed a hybrid hardware/software approach to address the mid-frequency LdI/dt issue. Their solution is meant to be an add-on to existing mid-frequency LdI/dt controllers. Their work performed dynamic optimization to alter instruction flows that induce large LdI/dt oscillation with compiler assistance, because purely hardware-based LdI/dt controller techniques incur a performance impact in the case of an emergency. They showed that software

pipelining, code motion, and instruction padding can modify program behavior that causes LdI/dt at the resonance frequency, while avoiding performance overheads.

While most of the previously mentioned works address the *mid-frequency* LdI/dt issue (50-100HMz), the focus of the work in this dissertation is to mitigate *high-frequency* LdI/dt that requires immediate response, for which the above solutions are inappropriate. Toward this effort, Powell and Vijaykumar proposed *Pipeline Muffling* [52], which is a technique that controls instruction issue and limits the use of resources to reduce high-frequency LdI/dt problems. Tang et al. [53] proposed controlled ramping of floating point units via scanning of the instruction fetch queue for upcoming instructions. This work has several limitations. First, it only deals with FPUs. Second, the LdI/dt effect of an instruction will vary depending on how it proceeds through different pipeline stages. Finally, like the other works in this area, neither the processor floorplan or power-pin locations are considered. Note that the cause of high-frequency LdI/dt is dependent on the spatial distribution of modules across the floorplan and their distances from the power-pins. In addition, high-frequency LdI/dt noise is not only dependent on a given module's self activity, but also correlated to gating events that stress nearby power-pins. None of the existing works accounted for this fact, which could result in current demand violations or false alarms. Unlike the work presented in this dissertation, none of these works combine microarchitectural techniques with physical design methods.

1.2.4 3D IC Design

The VLSI community has shown a great deal of interest in 3D ICs in the past. Multiple prior works have explored placement for 3D ICs [54, 55, 56, 23, 46, 57, 58], created wirelength prediction models to examine scaling trends for 3D ICs [59, 60, 61], performed 3D routing [62, 63], and optimized 3D via placement [64, 65]. The following subsections detail prior art specifically related to the work presented in this dissertation.

1.2.4.1 Floorplanning for 3D ICs

There have been several important works on 3D floorplanning. The first examination of the 3D floorplanning problem was provided by Deng and Maly [66]. They presented both a floorplanner and a placement engine that are 3D IC aware. Their proposed floorplanner operates on a simple 3D extension of the Bounded Slice-Line Grid [9] (BSG) floorplan representation, and uses simulated annealing for optimization. The simple extension is a multi-level BSG data structure with one BSG for each level of the chip stack. Deng and Maly [66] included neither thermal objectives nor the impact of TSV area in their work. Shiu *et al.* [67] examined multi-layer floorplanning for 3D SOPs, which contain active and passive components embedded in a multi-layer packaging substrate, and proposed an annealing-based 3D floorplanner capable of handling multiple constraints on module locations. However, Shiu *et al.* [67] required thermal and power-supply-noise issues to be encoded within the constraints input to their floorplanner, which is not appropriate for use in early-design-phase architectural exploration. Cheng *et al.* [68] presented an annealing-based floorplanner that operates on a novel slicing tree structure encoded using a static array. Their floorplan representation is capable of efficiently handling annealing moves under various module constraints. This work also requires that thermal constraints be input to their algorithm, which is inappropriate for early-design phase exploration. Cong *et al.* [24] presented a 3-D temperature-driven floorplanner based on the Transitive Closure Graph floorplan representation and a novel bucket structure to represent module overlap in the third dimension. They use various thermal analyzers to trade off run-time with accuracy and overall performance. However, this work does not consider early-design-phase exploration of microarchitectures and only presents results for simple benchmark circuits. The work presented in this disseration is the first to consider both performance and thermal objectives during microarchitectural floorplanning for early-design-stage exploration of high performance 2D and 3D processors.

1.2.4.2 Power-Supply-Network Design Techniques for 3D ICs

Previous work on 3D power delivery networks in 3D ICs has largely assumed a straightforward extension of 2D power delivery network design. Huang *et al.* [69] presented a physical model of 3D power distribution networks. In their model, power/ground through-silicon vias (TSVs) and power-supply C4 bumps are always aligned with one another. Jain *et al.* [70] extended the work of Gu *et al.* [71] by examining the use of multi-story power delivery in 3D ICs. In their approach there are two power domains and the ground network of one domain is the power network of the other domain. Again, the TSVs and supply bumps are always assumed to be fully aligned and they are divided among the three power distribution networks evenly. Yu *et al.* [72] demonstrated an optimization scheme for supply-bump assignment and TSV insertion simultaneously considering both supply noise and temperature. They again assume that supply bumps are aligned with TSVs in every case. In contrast, this dissertation proposes a 3D IC power-supply-network design technique that re-examines the unique capabilities of TSVs relative to package-level bumps.

1.2.4.3 Layout-Level 3D IC Physical Design Automation

There are a plethora of works presented in the literature that describe various 3D architecture design options and physical design algorithms for 3D ICs, but very few in the area of 3D design demonstration and methodology. Thorolfsson *et al.* [73] and Franzon *et al.* [74], the only prior works found, described the design of an FFT processor with 3D stacked memory implemented with MIT Lincoln Lab's 3-layer process [75]. However, they did not discuss cross-talk or power-noise analysis in 3D systems and did not include a thermal analysis. The work presented in this dissertation is the first to describe in detail a method for constructing and analyzing the physical layouts of a 3D processor using commercial tools from Cadence, Mentor Graphics, and Synopsys. Results related to 3D timing, power, thermal, IR-drop, signal integrity, and clock waveform analysis based on DRC/LVS-clean GDSII layouts are included.

CHAPTER 2

THERMAL-AWARE MICROARCHITECTURAL FLOORPLANNING

2.1 2D and 3D Thermal-Aware Microarchitectural Floorplanning

Power consumption and heat dissipation have become the dominant performance-limiting factors in modern day processors. These issues have limited the effectiveness of frequency scaling and have ushered in the multi-core era. Concurrently, improvements in process technology have increased the wire-delay-to-gate-delay ratio. In the future, continuing increases in this ratio will have a major negative impact on performance by causing processors to spend more time communicating than doing actual work. One way to combat wirelength issues is to move to 3D integrated circuits. By allowing circuits to overlap in the third dimension it is possible to significantly reduce wirelength. However, 3D ICs have to deal with increased power density and therefore even more thermal issues than processors produced with current technology. To effectively make use of 3D integration technology, a thermal-aware physical design tool is needed. Floorplanning is an important phase of the design flow in which to perform thermal optimization because it determines the locations of large collections of gates, and therefore has a significant impact on the overall power distribution of ICs. This chapter presents a thermal-aware floorplanner targeting microarchitectures.

2.1.1 Novel Design Flow

The novel design flow used in this work incorporates the dynamic power, leakage power, performance, and thermal analysis discussed in the next section. Figure 1 illustrates an overview of the design flow. First, technology parameters and an architectural description are used to estimate the area and delay of the microarchitectural modules. The analytical tools CACTI [76] and GENESYS [77] are used to estimate the area consumed by each module. Then, a cycle-level simulation using SimpleScalar [78] combined with Wattch [79]

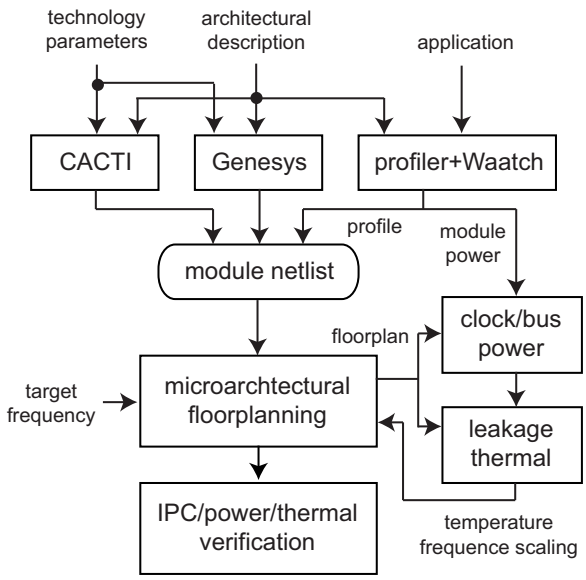


Figure 1. Overview of the microarchitectural floorplanning framework.

is done to estimate dynamic power consumption for each benchmark and collect and extract the amount of traffic between modules. From these tools a profile-weighted module-netlist and power consumption information are extracted and then fed into the multi-objective floorplanner. The clock power estimation from [80] and the leakage estimation from [81], as described in the next section, are also integrated with the thermal analyzer to guide optimization.

2.1.2 Modeling

2.1.2.1 Performance Modeling

High-frequency processors designed using deep-sub-micron technology will no longer be accurately simulated by architectural simulators that ignore inter-module communication latencies. These latencies result from wire delays, floorplan constraints, and thermal concerns. Both performance evaluation and floorplanning must take into account the inter-module latency, which is a function of distance, and the number of flip-flops between modules. In this work, the distances generated by the floorplanner are used to determine the latency-related parameters, such as pipeline depth and communication/forwarding latencies, used in performance simulation.

Ekpanyapong *et al.* [13] demonstrated that optimizing weighted wirelength based on the most highly used wires is a better heuristic for performance improvement than pure wirelength alone. This work uses the SimpleScalar [78] cycle-level microarchitectural simulator to collect the information needed to calculate these wire weights. Counters were added to the simulator to examine communication between architecturally-significant modules. The wire-access totals generated by the counters were normalized to produce the weights used on the wires during optimization.

2.1.2.2 Power Modeling

The power consumption profile for each microarchitectural module is generated while the inter-module traffic is collected. Power consumption profiles were gathered cumulatively for every 100,000 cycles and then averaged over all samples. The rationale for this style of sampling is that the temperature is very unlikely to elevate abruptly (within a few hundred thousand cycles) as a result of the thermal time constants of the constituent materials of integrated circuits and packages. For example, the thermal time constant of the POWER5 processor is on the order of 100's of milliseconds [18], while the cycle time for the same processor is 0.4 ns at minimum. The detailed traffic activity and dynamic power profiles are collected only once at the very beginning of the entire design flow.

The high operating frequencies of modern designs require that a large number of flip-flops be inserted into the clock distribution network. The large number of flip-flops result in a large load on the distribution network. The substantial percentage of the power budget that the clock distribution network consumes combined with the large number of flip-flops in modern designs necessitates modeling of the clock power at a finer granularity. Therefore, the accurate clock power model from Duarte *et al.* [80] is used for this work. The model considers clock distribution network power consumption for memory structure precharge arrays, distribution wiring and drivers, pipeline flip-flops, and the phase locked loop.

The leakage power is modeled in a separate process within the design flow. The model

is based on ChipPower [81], and considers different bias conditions, though it only estimates sub-threshold leakage power. For array-like structures, such as caches and TLBs, the number of bits (or SRAM cells) is multiplied by the amount of leakage current per bit and by the supply voltage to calculate the total leakage power for the structure. To calibrate the model used here, the sub-threshold leakage currents were also calculated using the method in eCACTI [82]. This model closely matches the leakage power estimated from eCACTI. For logic structures, CMOS gates are assumed where half the transistors are leaking at any given time. The area values from GENESYS [77] are used in estimating the number of transistors in these structures.

The leakage power is iteratively calculated with the temperature. Leakage power is first estimated based on an initial temperature calculation. Then this initial leakage power is used to calculate the temperature again. This process is repeated until convergence or thermal runaway is detected. The criteria from Liao *et al.* [83] are used for detecting thermal runaway: (i) the maximum module temperature, T_{max}, is increasing, or (ii) the positive change in power is larger than the package's heat removal ability. The package's heat removal ability is defined as $(T_{max} - T_a)/R_t$, where T_a and R_t are ambient temperature and thermal resistance of the package, respectively.

2.1.2.3 Thermal Modeling

The thermal model used in this work is based on the linearized differential equation ($k \cdot \nabla^2 T + P = 0$) for steady-state heat flow, as described by Tsai and Kang [19]. In the equation, T is the temperature, k is the thermal conductivity, and P is the power density of heat sources. The chip is divided into a 3D grid, as shown in Figure 2, to apply a finite difference approximation to the differential equation. The thermal equation is rewritten into the following matrix form: $\mathbf{R} \cdot \overrightarrow{P} = \overrightarrow{T}$, where $\overrightarrow{P}$ is the power profile vector (P_i is the power dissipation of node i), $\mathbf{R}$ is the thermal resistance matrix ($\mathbf{R}_{i,j}$ is the thermal resistance between node i and node j), and $\overrightarrow{T}$ is the temperature profile vector (T_i is the temperature of node i). Thus, a single matrix-vector multiplication can be used to calculate

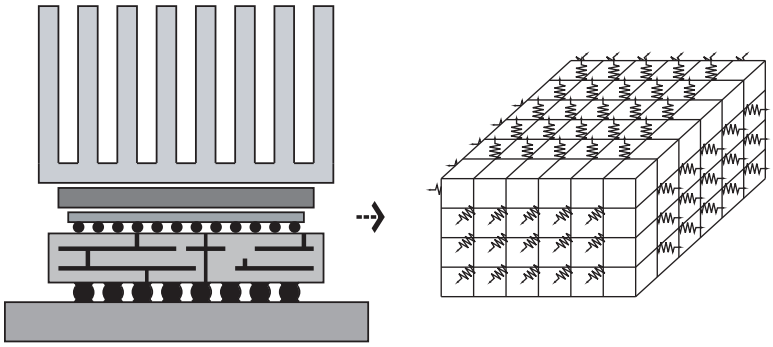


Figure 2. 3D grid of a chip for thermal modeling

the temperature of all nodes.

For the optimizations in this work a non-uniform 3D thermal resistor mesh was used to decrease execution times. Grid lines are defined at the center of each microarchitectural module for the *X* and *Y* directions and extend through the *Z* direction to form planes. The intersection of grid lines in the *X* and *Y* directions define the thermal nodes of the resistor mesh. Each thermal node models a rectangular prism of silicon that may dissipate power if it covers some portion of a block. The total power of each block is distributed according to and among the *X*-*Y* area of the nodes that block covers.

This thermal model is designed to provide fidelity for the optimization process, not accuracy, and to have short run-time. Hotspot [1] is a floorplan-level thermal simulator used in the microarchitecture community. Hotspot is not appropriate for use during optimization because of it's large run-time. However, Hotspot is used here to guage the accuracy of the non-uniform thermal model. Figure 3 provides a comparison of the temperatures calculated by the non-uniform model and those provided by Hotspot v3.0 across ten benchmarks on a single floorplan. The figure shows that the non-uniform model provides a similar temperature and provides fidelity between various power profiles.

2.1.3 2D Floorplanning Algorithm

The 2D floorplanning algorithm in this work is based on linear programming (LP). The basic idea behind the floorplanning algorithm is to perform recursive bipartitioning of the

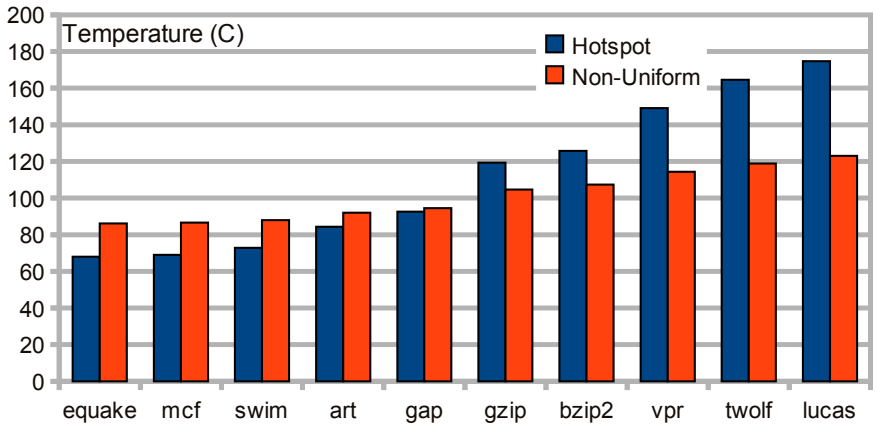


Figure 3. Comparison with Hotspot v3.0 [1]. Temperatures are in degrees Celsius.

modules until each partition contains a single module. The steps of the floorplanning process are show in Figure 4. In this approach, the overall relative location among the modules

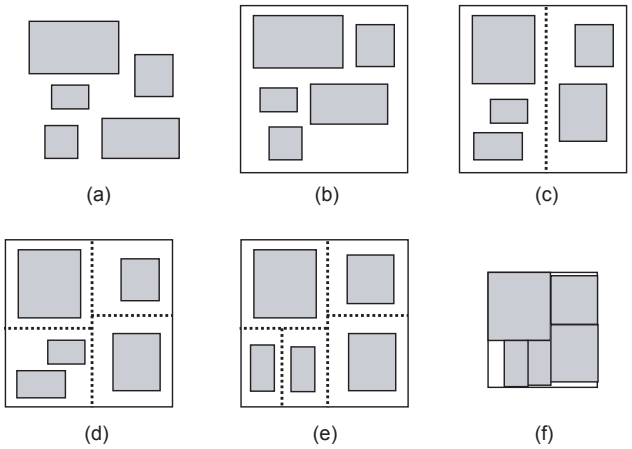


Figure 4. Illustration of 2D microarchitectural floorplanning. (a) A set of modules with size information (b-e) LP-based slicing floorplan, (f) non-slicing floorplan refinement.

is determined by the slicing operations and a separate LP determines the dimension of the modules and fine-tunes their locations. Once a partition has been chosen for division, the module temperatures are obtained by performing thermal/leakage analysis. Because there is no way to obtain block temperatures without a floorplan, the first iteration of the recursive bipartitioning contains no temperature objective. From then on the previous iteration's block positions are used to calculate the temperatures for the current iteration.

Linear-programming-based floorplanning is then used to simultaneously optimize the performance and temperature distribution under the target frequency, leakage, center of gravity constraints (to remove overlap among the modules), and boundary constraints.

A detailed explanation of the 2D and 3D LP formulation used in this work is included in Appendix A. The main difference between the 2D version of this formulation and that presented by Ekpanyapong *et al.* [13] is the addition of a temperature term to the objective function. The temperature term is formulated as $(1 - T_{ij})(X_{ij} + Y_{ij})$, where T_{ij} is the normalized product of the temperature of modules i and j, and X_{ij} and Y_{ij} are the distances between modules i and j in the x and y directions, respectively. This formulation was chosen as the temperature-dependent portion of the cost function because it satisfies several properties: It is linear with respect to distance between module i and module j, it considers the temperatures of both module i and module j, and it grows smaller when considering hot blocks, and larger when considering cool blocks. Because the cost function is being minimized in the LP and not maximized, it is necessary to only consider minimization of the distance between cool blocks and not maximization of the distance between hot blocks, as would be preferable.

There are several non-optimalities introduced by the standard LP relaxation of the floorplanning problem. The recursive bipartitioning process also yields only slicing floorplans. To address these issues, a simulated-annealing-based refinement engine was added as a post-process. A sequence pair is derived from the LP floorplanning result using the *gridding* algorithm described by Murata *et al.* [84], and low-temperature annealing is performed on that sequence pair. The following cost function is used during annealing:

$$cost = \alpha \cdot perf\_wire + \beta \cdot max\_temp + \gamma \cdot area$$

where $perf\_wire$ is the profile-weighted wirelength and max_temp is the maximum module temperature. The weighting constants, α and β, are the same as those used in the LP-based floorplanner.

2.1.4 3D-Specific Enhancements

Silicon dies that are vertically stacked require special kinds of vias for inter-die connections called through-silicon vias (TSVs). There are three kinds of TSVs depending on the bonding mechanism used to connect the two die together: *face-to-face* (F2F), *face-to-back* (F2B), and *back-to-back* (B2B) TSVs, as illustrated in Figure 5. The "face" refers

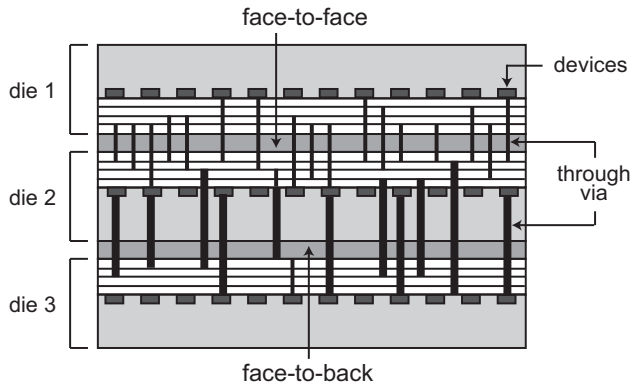


Figure 5. TSVs in 3D ICs with face-to-face and face-to-back bonding. Back-to-back style bonding occurs when the two substrate sides are attached (not shown in this figure).

to the metal layer side of a die, whereas the substrate side is called the "back." F2F TSVs ($\approx 0.5\mu \times 0.5\mu$) have a smaller pitch than F2B ($\approx 5\mu \times 5\mu$) and B2B TSVs ($\approx 15\mu \times 15\mu$) [85]. It is desirable to *reduce* the number of inter-die connections in F2B/B2B bonding. In the case of F2F bonding, however, it is desirable to *increase* the number of inter-die connections since the via density is much higher (almost the same as the intra-die via density) and therefore enables a significantly higher bandwidth for inter-layer communication.

A two-step approach to 3D floorplanning is used in this work. Modules are first partitioned into layers and then these layers are simultaneously floorplanned. The goal during layer partitioning is to exploit the bonding style of the manufacturing process. This is accomplished using the FLARE partitioner [86]. During layer partitioning, a layer is assigned to each module such that the number of connections at the F2F boundary is maximized, while the number of connections at the F2B/B2B boundary is minimized. First, the modules are partitioned into two groups with minimum profile-weighted cut size. Next, pairs of modules within those groups connected by high profile-weighted edges are split into two

layers with F2F bonding so that they can take advantage of shorter interconnects to achieve better performance.

The LP-based 2D floorplanning algorithm is extended handle 3D floorplanning by simultaneously handling multiple layers. Specifically, each slicing cutline is inserted to cut all layers simultaneously, as illustrated in Figure 6. The goal of slicing 3D floorplanning remains the same as the 2D case: to determine the dimension and relative position among the modules so that the multi-objective cost function is minimized. In addition, these locations are refined via 3D non-slicing floorplanning during the annealing-based post refinement. The major difference between the 2D and 3D slicing floorplans are the interaction with different layers, which is a key element for an effective 3D floorplan. More specifically, area optimization has to be footprint-aware; area increases from the smallest layer can be easily tolerated because those increases are unlikely to increase the overall footprint area. The LP formulation reflects this new optimization goal, which is unique to 3D floorplanning. Since layer partitioning has already addressed issues related to bonding style, the modules are not allowed to move to other layers during the bipartitioning process. This limitation is partly based on constraints of the linear objective function that is required during LP optimization.

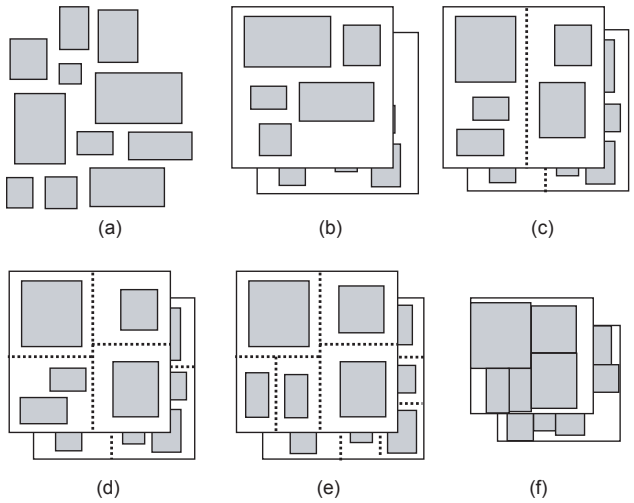


Figure 6. Illustration of 3D microarchitectural floorplanning. (b) layer partitioning, (c-e) LP-based 3D slicing floorplan, (f) non-slicing floorplan refinement.

The goal of the 3D stochastic refinement process is to improve the 3D slicing floorplan obtained from the LP-based construction algorithm. A stochastic refinement process is used because the LP-based floorplans were found to have unnacceptably large whitespace. The basic approach is the same as the 2D case: non-slicing floorplanning with low-temperature simulated annealing to simultaneously refine the performance, temperature, and area objectives. The major difference between the 2D and the 3D case is that one sequence pair per layer is used to represent the entire 3D floorplan. In addition, the perturbation scheme does not allow inter-layer module movement to maintain the bonding-aware layer separation and remain close to the minima found by the LP solver.

2.1.5 Experimental Results

Ten programs from the SPEC2000 benchmark suite were chosen for the experiments in this work. Four were chosen from the floating point and six from the integer benchmark suites. For IPC evaluation, each benchmark was run on the average-case floorplan using a version of SimpleScalar 3.0 [78] modified as discussed in Subsection 2.1.2.1. The IPC of each benchmark was calculated by fast-forwarding 4 billion instructions and gathering performance statistics on the next 4 billion instructions. The reported temperature is simulated after all floorplanning steps and is adjusted relative to a $45^{\circ}C$ ambient temperature. The 3D floorplanning optimizations were performed on a four-layer stacked IC. Face-to-face bonding is assumed between layers 0 (bottommost) and 1 and between layers 2 and 3. Back-to-back bonding is used between layers 1 and 2. The heat sink is attached to layer 3. Wirelength is reported in millimeters. The "area" in these results refers to the footprint area (maximum width $\times$ maximum height) of the four-layer floorplan and is reported in millimeters squared.

Tables 1 and 2 present a comparison of the performance (P), temperature (T), area (A), wirelength (W), and execution time of four different objective functions for the 2D and 3D cases. All data in these tables are taken from the combined LP+SA approach. A major assumption of this work is that dynamic power dissipation does not change significantly due

Table 1. Multi-objective 2D floorplanning results with performance (P), maximum block temperature (T), area (A), wirelength (W), and execution time reported. Temperature is in degrees Celsius. Whitespace (WS) is reported as a percentage.

	2D floorplan							
	A+W		A+P		A+T		A+P+T	
bench	IPC	temp	IPC	temp	IPC	temp	IPC	temp
gzip	2.01	78.3	2.83	100.4	2.03	75.2	2.69	86.2
swim	0.52	64.3	0.85	78.4	0.54	63.0	0.66	70.5
vpr	0.95	87.6	1.19	113.8	0.82	82.3	1.15	95.9
art	0.38	67.9	0.62	83.3	0.39	65.4	0.51	74.4
mcf	0.07	63.0	0.09	76.9	0.07	62.1	0.10	69.4
equake	0.40	62.7	0.47	76.3	0.41	61.8	0.43	69.0
lucas	0.63	95.6	0.75	123.2	0.64	88.3	0.80	103.5
gap	1.17	70.1	1.24	87.8	1.18	68.1	1.32	77.3
bzip2	1.42	80.4	1.90	103.6	1.47	77.1	1.65	88.4
twolf	0.60	92.3	0.94	120.8	0.61	85.8	0.61	101.1
AVG	0.81	76.2	1.09	96.46	0.82	72.9	0.99	83.6
AREA (mm^2)	52.46		57.23		58.66		60.37	
WIRE (mm)	345.20		412.15		358.86		449.67	
TIME (sec)	174		188		1116		1064	
PIPE	22		19		27		23	
WS (%)	10		20		23		21	

to changes in module position. All variations in temperature among the experiments are due to thermal coupling and changes to the clock power, bus power, and leakage power because the dynamic power consumption is not re-simulated during the evaluation phase. For the 2D case, the maximum module temperature increased markedly for the A+P objective compared to the baseline A+W objective. The IPC result of A+P is the best among the four algorithms, with an average IPC improvement over A+W of 35%. The A+T objective decreases the temperature by about 24% over the A+P objective, while the IPC decreases by 25%. The hybrid A+P+T objective decreases the temperature by 14% over the A+P objective while maintaining a high IPC value of 22% above the baseline A+W objective.

For the 3D cases, 3D the A+W objective achieves a 37% increase in IPC and a 34% increase in temperature over the 2D A+W objective, while decreasing the total wirelength by almost 40%. The area result of the 3D A+W objective is the best among all objective functions. The A+P objective increases the IPC by 18% over the A+W objective and

Table 2. Multi-objective 3D floorplanning results with performance (P), maximum block temperature (T), area (A), wirelength (W), and execution time reported. Temperature is in degrees Celsius. Whitespace (WS) is reported as a percentage.

	3D floorplan							
	A+W		A+P		A+T		A+P+T	
bench	IPC	temp	IPC	temp	IPC	temp	IPC	temp
gzip	2.74	104.7	3.98	125.9	2.75	98.9	2.85	104.7
swim	0.71	92.9	0.85	106.9	0.72	84.1	0.92	88.0
vpr	1.30	111.5	1.40	137.0	1.25	107.1	1.29	114.4
art	0.52	95.6	0.59	111.4	0.52	87.9	0.61	92.0
mcf	0.10	92.0	0.11	105.4	0.10	83.1	0.07	86.6
equake	0.54	91.7	0.58	105.0	0.55	82.6	0.67	86.2
lucas	0.87	116.9	0.92	145.3	0.88	113.0	1.19	123.0
gap	1.59	97.0	1.59	114.2	1.62	89.6	1.61	94.5
bzip2	1.94	106.8	2.05	129.0	1.98	101.5	2.33	107.4
twolf	0.81	114.6	1.03	142.2	0.84	111.0	1.02	118.9
AVG	1.11	102.4	1.31	122.2	1.12	95.8	1.26	101.6
AREA (mm^2)	22.20		23.63		25.45		26.45	
WIRE (mm)	217.20		323.43		252.08		247.02	
TIME (sec)	180		438		16913		20016	
PIPE	22		17		24		21	
WS (%)	9		16		25		23	

increases the temperature by 19%. As expected, the A+T objective decreases the temperature result compared to the A+P objective significantly and achieves the best temperature results among all four 3D algorithms. The temperature simulations in the 3D case require a grid size four times larger than for the 2D case, which causes the execution time of those objectives incorporating temperature calculations to increase dramatically. The hybrid A+P+T objective retains a temperature close to that of the A+W objective while increasing the IPC by 14%. In summary, the A+P+T objective (i) obtains results that are between those of the A+T and A+P objectives and (ii) outperforms the A+W objective in terms of performance with comparable temperature results for both 2D and 3D. If the temperature objective should be emphasized, the thermal weight can be increased, which will likely lead to further performance degradation.

A trade-off between performance and temperature is shown in Figure 7. Temperature and IPC are reported as averages over the 10 benchmarks. The performance and area weights are held constant while the thermal weight is varied. The graph shows that when

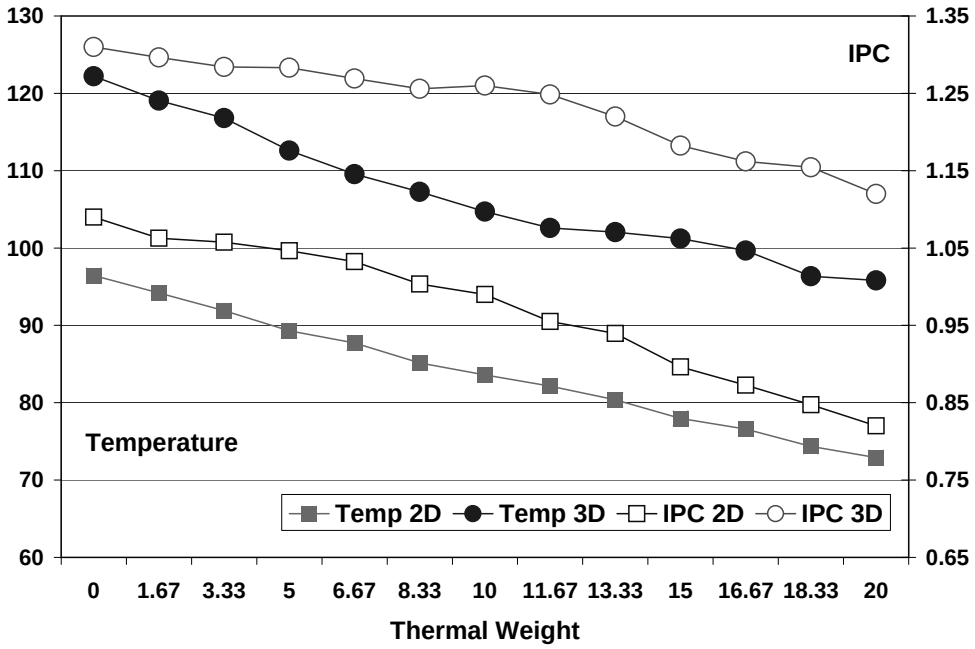


Figure 7. A trade-off between performance and temperature. Performance and area weights are held constant while thermal weight varies.

the thermal weight is given more consideration by the floorplanner, the performance drops. Ideally there would be some separation between the curves to indicate that high reduction in temperature could occur with little degradation in IPC value. The sweet spot of the curve appears when the thermal weight is around 10. The IPC drops sharply after this and so would be undesirable for the reduction in temperature achieved. One can observe that there is a 15% reduction in IPC and a 22% reduction in temperature between the performance-only objective (0) and the highest-weight hybrid objective (20) for the 3D case. As expected and also shown in Tables 1 and 2, the multi-layer floorplans increase both the temperature and IPC over the single-layer floorplans. Also of note is that the highest-thermal-weight multi-layer floorplan has a temperature close to that of the lowest-thermal-weight single-layer floorplan while achieving a higher IPC. This demonstrates the benefits rendered from the move to multi-layer ICs.

2.1.6 Summary

This chapter presents the first multi-objective microarchitectural-level floorplanning algorithm for high-performance high-reliability microprocessors targeting both 2D and 3D ICs.

We simultaneously consider performance and thermal objectives such that our automated floorplanner can provide a balanced or goal-directed processor organization that achieves user-specified design objectives. Moreover, we integrate leakage modeling into our thermal analyzer and monitor the temperature/leakage interaction to prevent thermal runaway. We investigate vertical overlap among the modules in 3D floorplanning and how it affects the performance, thermal, and area objectives. In addition, we partition the modules into multiple layers while considering TSV requirements for F2F and F2B bonding styles. Our hybrid approach that combines linear programming and simulated annealing optimization proves to be very effective in obtaining high-quality solutions.

2.2 Multi-Granularity Multi-Core Microarchitectural Floorplanning

This section develops a thermal-aware multi-core microarchitectural floorplanner. In recent years the multi-core paradigm has become the *de facto* standard in general-purpose processor design. The quick pace of the shift to the multi-core paradigm has mainly been driven by the need for low power density and high complexity effectiveness. High power densities cause increases in average and maximum operating temperatures and associated decreases in reliability. Floorplanning is useful in the multi-core context for reducing thermal problems, much as it is in 3D floorplanning. In multi-core systems, multiple cores are placed on the same chip. Thermal coupling between modules on the edges of neighboring cores can lead to serious thermal problems. Figure 8 shows an example of thermal coupling. The floorplan on the left of Figure 8 is tiled horizontally to create the floorplan on the right. The maximum temperature seen in the floorplan is then increased from 86.1°C to 99.8°C.

2.2.1 Terminology

The terminology used in this chapter may be somewhat confusing because of the vague definitions of terms like block, module, and core. Therefore, a summary of the terms used in this chapter, and their meanings, is given in Table 3. Module planning is the only optimization used in homogeneous floorplanning and fixed heterogeneous floorplanning.

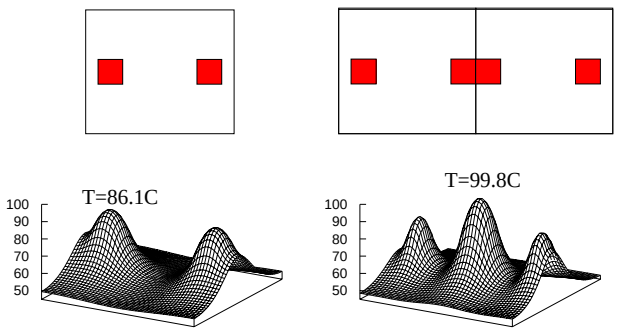


Figure 8. The effect of core tiling on the thermal profile. The per-core power dissipation remains constant, but thermal coupling causes the maximum temperature to be higher for the tiled floorplan.

Table 3. A summary of the terms used in this chapter.

term	**explanation**
moduleplan	arrangement of modules in a single core
coreplan	arrangement of cores and cache banks
multi-core floorplan	arrangement of modules and cores (moduleplan plus coreplan)
homogeneous coreplan	coreplan with only one type of moduleplan
heterogeneous coreplan	coreplan with multiple moduleplans
fixed heterogeneous coreplan	hand-optimized heterogeneous coreplan with no split cache
multi-granularity floorplanning	optimizing both module- and coreplans

The six types of fixed heterogeneous coreplans considered in this chapter are shown in Figure 9.

2.2.2 Moduleplan Optimization

The moduleplanning algorithm presented in this chapter uses simulated annealing with the Sequence Pair [8] representation. Three move types are used during annealing. The first move swaps the position of two modules in both sequence pairs. The second move swaps the position of two numbers in a single sequence. The final move changes the shape of one module [87]. A single sequence pair is sufficient for the purpose of optimizing moduleplans in homogeneous coreplans. Conversely, k separate sequence pairs must be managed for heterogeneous coreplans with k types of cores. During the annealing process, a core type is first selected for performing a candidate move on. Next, the move type is selected for that core type. This combination allows the simultaneous optimization of all moduleplans

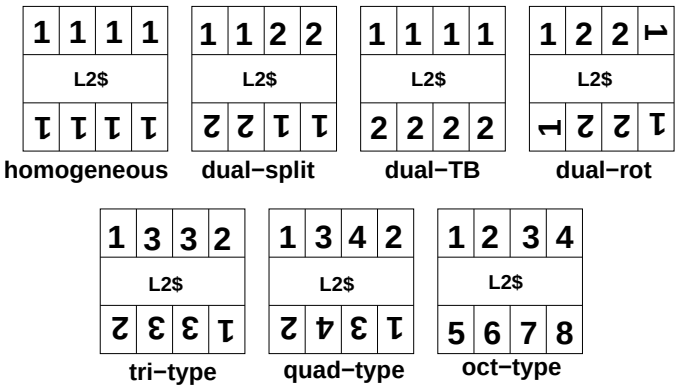


Figure 9. Homogeneous vs. fixed heterogeneous coreplans for an eight-core system. Module planning is the only optimization used on these fixed coreplans.

in the heterogeneous coreplan.

The objectives of the annealing-based optimization are to increase performance and decrease maximum and average temperatures. Performance is optimized by using profile-weighted wirelength [13] as a part of the cost function. The cost function optimized during moduleplanning is:

$$Cost = \alpha * area + \beta * wwl + \gamma * temp$$

where α, β, and γ are user-defined weighting constants, *area* is the total area of the coreplan, *wwl* is the sum of the profile-weighted wirelength of all core types in the system, and *temp* is related to maximum and average temperatures, as described next. To lower operating temperatures, a temperature-weighted block-distance cost, *temp*, is added to the annealing cost function. The addition is calculated as the temperature-weighted distance between each pair of modules in the floorplan. Calculating the temperature at every annealing move is too costly in terms of execution time. Therefore, the temperature is updated only every 20 annealing moves. Experiments were used to demonstrate that this limit is small enough to generate good solutions and large enough to give reasonable execution time.

2.2.3 Coreplan Optimization

Coreplanning involves optimizing the location and orientation of cores and cache banks. Compared to pure moduleplanning, the additional complication of coreplan moves comes into play during coreplanning. If unrestricted coreplan moves were allowed during annealing, then a large number of highly unacceptable coreplans would be examined. Coreplans that have high aspect ratios and large amounts of whitespace cover a large portion of the solution space. A fixed coreplan space that consists of a square grid is used to combat this issue and reign in execution time. Specifically, a 4×4 grid is used for the eight-core system with eight L2-cache banks examined in this section. Each point on the grid may be occupied by either a core or a bank of the L2 cache. Each core and bank is also associated with an orientation. The "top" of the core or cache is the connection point for the bus that connects all the cores to each bank.

During coreplanning, there are two types of moves allowed on the fixed grid. The first move swaps the location of two objects (either a core or a bank) at two randomly chosen grid points. The second move rotates the object at a randomly chosen grid point to another orientation. The temperature is recalculated after every coreplan move because of the large changes that occur when a coreplan move is performed.

2.2.4 Floorplanning Algorithm

Multi-core floorplanning in this paper refers to the combination of both moduleplanning and coreplanning. The goal of multi-core floorplanning is to simultaneously optimize the module and core locations to maximize the performance and minimize the maximum temperature. The two different move granularities used in this floorplanner impact the cost function in different ways. For example, a coreplan move will not change the area of each of the moduleplans or the weighted wirelength within each moduleplan. The floorplanner exploits this difference by calculating the area and wirelength objectives only once if a set of coreplan moves is going to be evaluated to save execution time. However, if the coreplan and moduleplan moves are mixed together, then the full set of costs must be recalculated

for each move.

2.2.4.1 Static Multi-Granularity Floorplanning

There are several options to consider when combining coreplanning and moduleplanning moves. One option is to consider them to be equivalent moves and allow either to be randomly selected during annealing. However, this option ignores the fact that moving a core to a different section of the chip will completely change its thermal environment. Using this option would require careful selection of a temperature update policy and the ratio of selection between coreplan and moduleplan moves. This method is referred to as the *Random* method.

A second option is to perform a single coreplan move followed by a large number of moduleplan moves. The coreplan move is either accepted or rejected and then the moduleplan moves begin. The moduleplan moves are then accepted or rejected based on the cost function as they are made. The most obvious issue with this method is that it limits the amount of optimization that each coreplan is allowed to receive before the next coreplan is evaluated. This can result in inefficient optimization speed. This method is referred to as the *Inner Loop* method.

The final non-adaptive method analyzed in this section uses a number of coreplan moves followed by a number of moduleplan moves. For every annealing temperature-level a number of coreplan moves are first accepted or rejected and then a number of moduleplan moves are accepted or rejected. This method is referred to as the *Two Sets* method.

2.2.4.2 Adaptive Multi-Granularity Floorplanning

Additionally, adaptive methods may be used to intelligently choose between the two available move types. The first adaptive method evaluated in this section springs from medium access control methods used in networking. This method relies on credits to determine the success rate of each type of move. Each type of move (coreplan and moduleplan) is initialized with a number of credits at each annealing temperature level. Whenever a move is completed, the number of credits for that move type is increased if the move is accepted

or decreased if the move is rejected. When a move type runs out of credits, then the other type of move is selected instead. Effectively, when one type of move is doing badly, the algorithm switches to the other type. The number of credits that are given to each move type is based on the acceptance percentage of the previous annealing temperature level. When more moves are accepted the number of credits given is low; when fewer moves are accepted the number of credits given is high. This method is referred to as the *Adaptive Credits* method. Pseudocode for this method is shown in Figure 10.

```
 1: for all annealing temperature levels do
 2:     initialize floor and core credits;
 3:     if previous move type has credits left and has
        moves left then
 4:         choose previous move type;
 5:     else if other move type has credits left and
        has moves left then
 6:         choose other move type;
 7:     else
 8:         reinitialize credits;
 9:     end if
        do selected move
10: end for
```

Figure 10. Pseudocode for the Adaptive Credits coreplanning method, which uses a credit-based scheme.

The second adaptive method that is considered is based on the Inner Loop fixed floorplanning method described above. A limit for the number of bad moduleplanning moves per coreplan move is allocated for each annealing temperature level. This limit is again based on the previous annealing-temperature-level's acceptance percentage. For a high acceptance percentage, a small limit is used. For a low acceptance percentage, a large limit is used. Whenever this limit of bad moduleplan moves is reached, the inner loop ends and the next coreplan move is attempted. The intuition used here is that a large number of bad moduleplan moves may be indicative of larger problems at the coreplan level. This method is referred to as the *Adaptive Loop Limit* method. Pseudocode for this method is shown in Figure 11.

```
 1: for all annealing temperature levels do
 2:     for i = 0 to number of coreplan moves do
 3:         initialize number of allowed bad floor-
            plan moves;
 4:         do coreplan move;
 5:         accept/reject coreplan move;
 6:         for j = 0 to number of floorplan moves
            do
 7:             do floorplan move;
 8:             accept/reject floorplan move;
 9:             if number of rejected floorplan moves
                > number of allowed bad floorplan
                moves then
10:                 exit floorplan move loop;
11:             end if
12:         end for
13:     end for
14: end for
```

Figure 11. Pseudocode for for the Adaptive Loop Limit coreplanning method. One coreplan move is followed by a number of floorplan moves that is limited to specified value.

2.2.5 Bus Routing

One issue that is unique to coreplanning is dealing with the memory bus that connects the cores and the cache banks. In a fixed heterogenous coreplan, the bus length will change only in a minor way during optimization as a result of changes in the locations of the first level caches; conversely, moving cores and caches around can change bus length drastically. Because the bus is a major architectural issue, it must be considered during the coreplanning process. Changes in bus latency and congestion can have a major impact on the performance of multi-core systems. Because there is only a small number of memory buses in the banked system, a simple router [88] is used to perform full bus routing after each annealing move. The bus length is added to the annealing cost function to minimize routing requirements and bus latency.

2.2.6 Experimental Results

Select programs from the SPLASH-2 [89] parallel benchmark suite are used for evaluation in this section. The benchmark applications are simulated on the SESC [90] multi-core

simulator. The architecture used for these experiments is summarized in Table 4. The IPC

Table 4. The architecture used in the experiments.

Parameters	Values
Number of cores	8 cores
Clock frequency	5 GHz
Fetch width	4-wide
Issue/Commit width	4-wide
branch predictor	Combining: 16K entry metatable Bimodal: 16K entries 2-Level: 11 bit BHR, 16K entry PHT
BTB	2-way, 2048 sets
L1 I- and D-cache	32KB 2-way 32B line
I- and D-TLB	64 entry
L2 cache	8MB 8-way unified 32B line
L1/L2 latency	2 cycles / 12 cycles
Main memory latency	500 cycles
LSQ size	64 entries
ROB size	176 entries
Functional units	4 int ALUs, 2 FP ALUs

value for a benchmark is calculated by adding together the IPC values of each core in the multi-core architecture over an entire benchmark run to represent the overall throughput. The IPC value for a multi-core floorplan is calculated by taking the geometric mean of the IPC values obtained for each benchmark.

The benefits of the most basic move to multi-core floorplanning are examined in this section. In traditional-style homogeneous moduleplanning the temperature is evaluated in a single-core environment. In multi-core-aware homogeneous moduleplanning the temperature is evaluated in a multi-core environment. Traditional-style moduleplanning ignores heat coupling with neighboring cores. Traditional-style moduleplanning is the cheapest possible option to implement multi-core systems using off-the-shelf cores. A comparison of the maximum and average temperatures between traditional-style and multi-core-aware moduleplanning is presented in Figure 12. IPC results for the same set are given in Figure 13.

The IPC results of multi-core-aware moduleplanning are nearly the same as those of

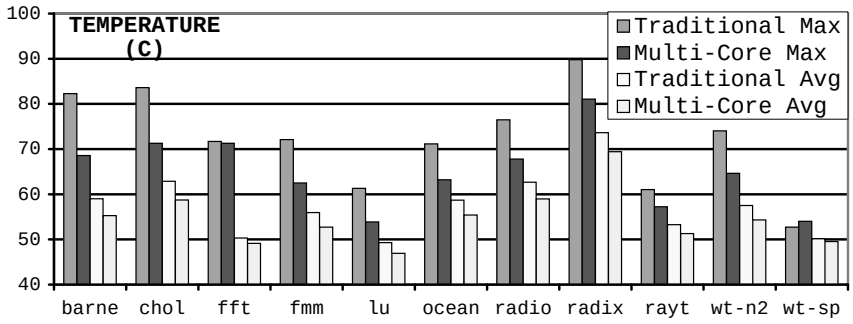


Figure 12. A temperature comparison between traditional-style and multi-core-aware homogeneous moduleplanning

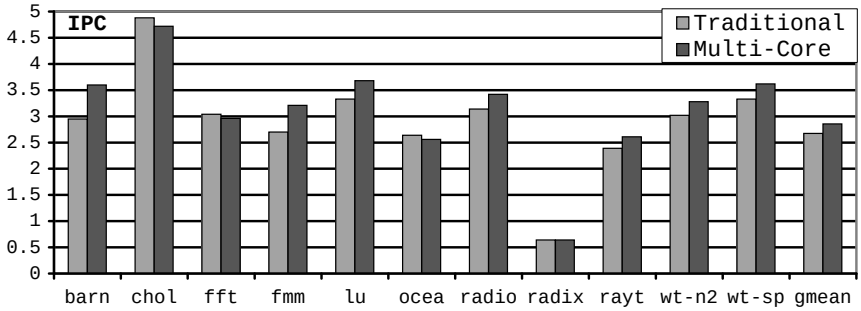


Figure 13. An IPC comparison between traditional-style and multi-core-aware homogeneous moduleplanning

the traditional-style moduleplanning. However, the maximum temperature for the multi-core-aware moduleplanning is 10% lower and the maximum average temperature is 6% lower. A comparison of the temperature profiles of a *single core* from both floorplans, shown in Figure 14, explains the improvement. Traditional-style moduleplanning produces

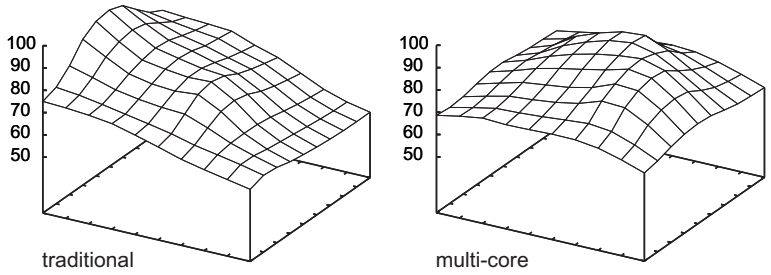


Figure 14. A temperature profile comparison between a single core of traditional-style and multi-core-aware moduleplanning. The traditional-style moduleplan has high temperatures near the core boundaries. Thermal coupling causes the average and maximum temperature to be higher.

a temperature distribution that has hot modules near the chip boundaries. In multi-core-aware moduleplanning, the neighboring cores will tend to push their hot modules away

from the boundaries of the chip. Figure 15 shows the moduleplan of the single cores used

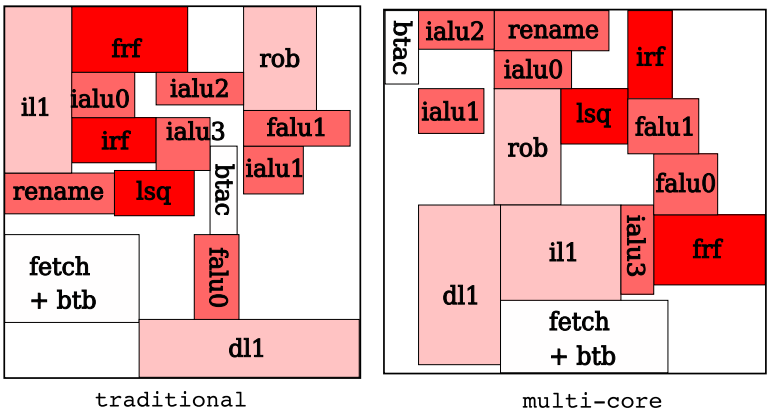


Figure 15. Moduleplans for the cores in Figure 14.

in the temperature profiles of Figure 14. The distribution of the ALUs and the register files demonstrate some of the reasons for the temperature profile differences between the two approaches. These results emphasize that multi-core-aware moduleplanning is better than traditional-style moduleplanning for homogeneous (single core-type based) multi-core systems.

Next, a comparison of the six fixed heterogeneous coreplan types shown in Figure 9 is presented. For the purpose of fairness among the different numbers of core types, each coreplan type is given the same optimization time budget. For example, an optimization using two core types would be given twice the total time budget of an optimization with a single core type. The results from traditional-style homogeneous moduleplanning and multi-core-aware homogeneous moduleplanning, as presented in Section 2.2.2, are included for comparison. The baseline is traditional-style homogeneous moduleplanning.

Figure 16 shows the IPC and temperature comparison results. As expected, with largest design effort comes the best results and the 8-type coreplan has the best temperature values. The largest drop in temperature values occurs with the move from two core types to three core types. The IPC of the different coreplans varies due to thread-to-core assignments that are not controlled for in this section, as well as moduleplan differences that induce variation in microarchitectural delays. No pattern in the IPC values is readily discernible. Though

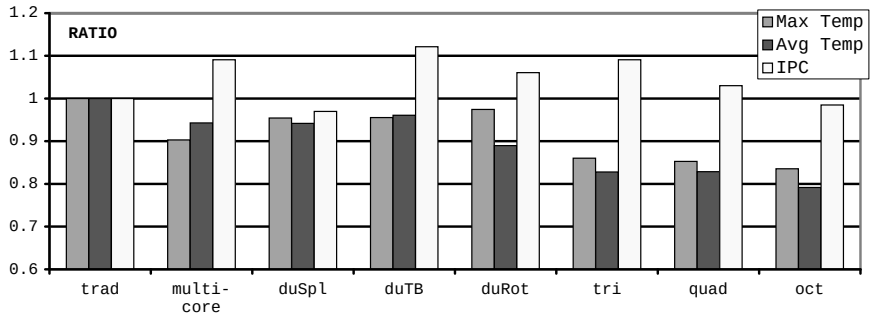


Figure 16. A comparison between several fixed heterogeneous coreplan types.

these coreplans conflict with the decreased design effort benefit of the move to multi-core, they show that there is some temperature benefit to be gained.

Next, a comparison of the results among the five different styles of multi-granularity floorplanning discussed in Subsection 2.2.4 is given. The three fixed and two adaptive algorithms were run with one type of core in the coreplan (homogeneous style). The IPC and maximum and average temperature of the results for each algorithm are shown in Figure 17.

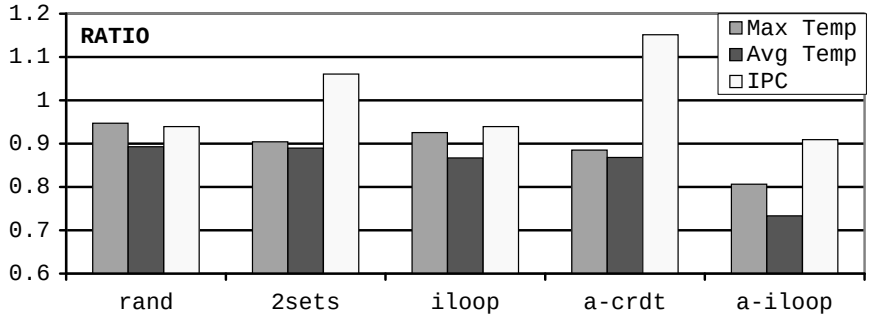


Figure 17. A comparison among the 5 multi-granularity floorplanning algorithms.

The lowest temperatures are achieved using the Adaptive Loop Limit method, which provides a 19% and 27% drop in maximum and average temperature, respectively, when compared to the traditional-style single-floorplan results. Because the Adaptive Loop Limit method provides the best temperature results, all subsequent experiments use this method. An example floorplan and the resulting bus routing from the Adaptive Loop Limit method are shown in Figure 18.

Finally, a study of the impact of floorplanning with cache banks is given below. The

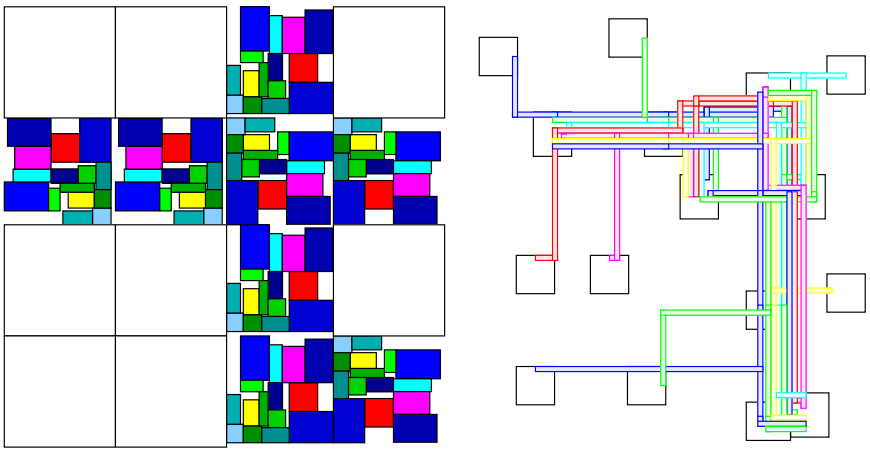


Figure 18. An example of a single core-type floorplan from the Adaptive Loop Limit method, and the resulting memory bus routing.

L2 cache is partitioned into eight banks so that the area of each bank is similar to that of a core. The eight cores and eight banks can be easily arranged into a 4×4 grid because each bank and core occupy similar area. The Adaptive Loop Limit method is used to optimize the floorplan (both moduleplan *and* coreplan) of these 16 blocks. Two hand-optimized coreplans, named *edges-type* and *checker-type*, as shown in Figure 19, are used for comparison. Figure 20 shows a comparison of the IPC and average and maximum

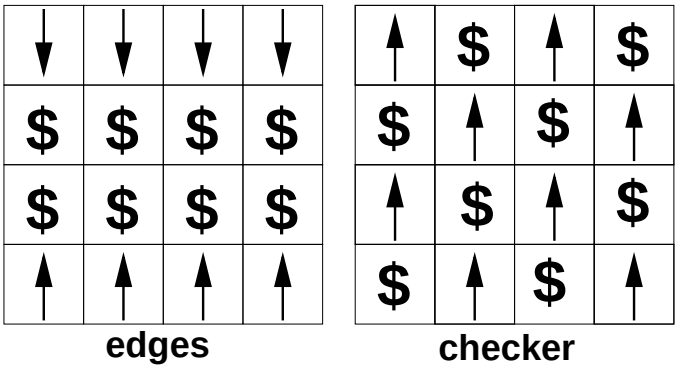


Figure 19. Hand-optimized coreplans with cache banks.

temperatures. For this comparison, the Adaptive Loop Limit method is based on a single core type (homogeneous). The hand-optimized edges-type and checker-type coreplans are not capable of achieving the same performance and thermal results as the Adaptive Loop Limit method. The IPC of the three methods again varies somewhat; however, this is due

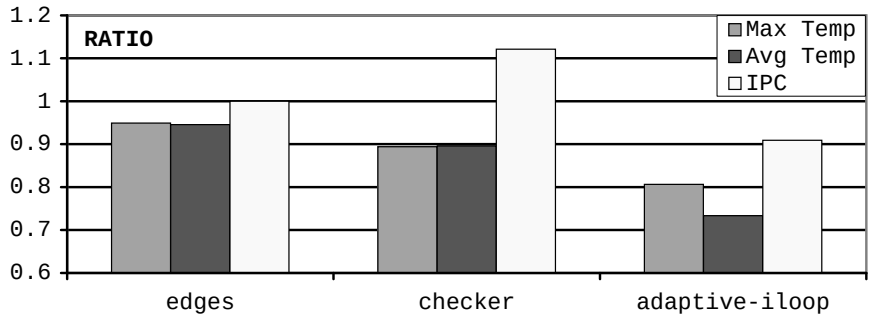


Figure 20. A comparison of the IPC and average and maximum temperatures among edges-type, checker-type, and the Adaptive Loop Limit algorithm.

to random variation in the floorplans and not a difference resulting from optimization.

Now we provide a discussion of the results for multiple core-type floorplaning with cache banks. Comparisons were made among designs with 1, 2, 3, 4, and 8 independent core types within a coreplan. Figure 21 shows the IPC and average and maximum temperatures for the various floorplanning options. Once again, more design effort yields

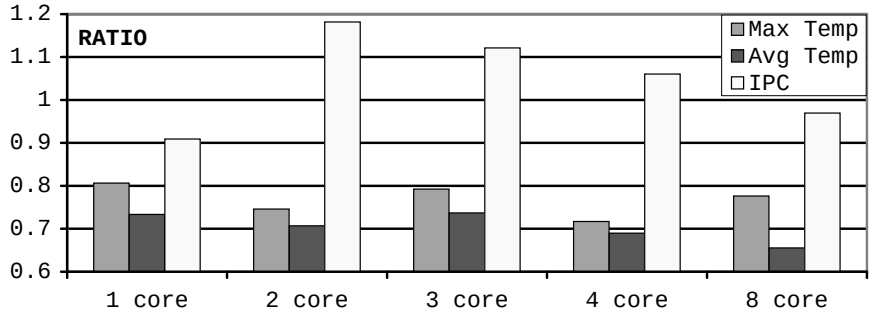


Figure 21. A comparison of the IPC and average and maximum temperatures among Adaptive Loop Limit method with various numbers of core types.

better results, with the higher number of core types yielding the lowest temperatures. At this level, however, the available optimization room is rather small. To ensure a fair comparison between types, equal time budgets were given based on the number of core types. This results in extremely long execution times for the optimizations with larger numbers of core types. For example, the eight-core optimization averaged a 50 hour execution time. However, the results show that there is some room for thermal improvement.

2.2.7 Summary

Design for multi-core systems presents a new challenge for the electronic design automation community. Early phases of the design cycle, such as microarchitectural-floorplanning, must take into account many factors. This chapter presents a microarchitectural floorplanner targeted at multi-core systems. Our methodology successfully reduces operating temperatures while maintaining system performance by taking into account module-to-module interatctions with neighboring cores. In the simplest approach our multi-core design achieves a 10% decrease in maximum temperature and a nearly 6% decrease in average temperature with a negligible change in IPC relative to tradtional-style floorplanning.

Our Adaptive Loop Limit multi-granularity multi-core floorplanning method achieves a 19% reduction in maximum temperature and a 27% reduction in average temperature compared to traditional single-core floorplanning. We also present results for heterogeneous floorplan types in a multi-core system with homogeneous architecture. All improvements resulting from the algorithm presented in this chapter come without physical-aware scheduling algorithms or dynamic temperature control schemes.

CHAPTER 3

POWER-SUPPLY-NOISE-AWARE FLOORPLANNING WITH DYNAMIC NOISE CONTROLLER

High-performance, power-conscious microprocessors exhibit varying current demands depending on the execution characteristics of a given program. For a high frequency microprocessor, any abrupt change in current demand (referred to as dI/dt) will result in high-frequency inductive noise that leads to voltage ringing in the power-supply network. In the worst case, unreliable supply voltage can flip data values, resulting in incorrect computation. Processors are often over-designed with the use of excessive amounts of decoupling capacitors (decap) to address this reliability issue. For increasingly complex processors, inserting excessive amounts of decap increases the chip area and at the same time exacerbates the leakage power problem. Moreover, significant design effort and cost for a worst-case design is inevitably expended to manage the infrequent cases where programs exhibit the maximum swing in current demand during the course of execution.

Traditional technology scaling has exacerbated the problem as well. Decreasing supply voltages reduce the absolute noise margin, and increasing transistor counts and higher frequencies result in more power dissipation. Aggressive power-saving techniques like clock gating and power gating have been widely studied and applied. Processors such as the Intel Pentium 4, Pentium M, and IBM Power5 [91] use different levels of clock gating to dynamically disable portions of the circuit. The industrial community has acknowledged the dI/dt issue with the extensive application of clock gating and responded with architectural solutions. For instance, the level two cache in the Power5 processor uses progressive clock gating in different cache banks to mitigate the dI/dt effect [91].

Conventionally, the worse-case current consumption is profiled and gauged by exercising power virus programs [42]. These programs are written expressly to vary the execution behavior of the microprocessor to induce drastic current-demand fluctuations. Designers

then allocate an appropriate amount of decap to manage these worst-case fluctuations in a repetitive process until the reliability targets are met. The main drawback of this design technique is that a significant amount of chip area is devoted to these decaps, which only cover infrequent worst-case behavior. For example, the Alpha 21264 reported that roughly 15 to 20% of the die area is occupied by decaps [92].

To address these shortcomings in the worst-case design methodology, we advocate a design procedure that builds awareness of inductive noise into the complete processor design process. Our proposed methodology involves two components. The first component is a noise-aware floorplanner that uses microarchitectural feedback. With this floorplanner we introduce the following innovations:

- Two metrics called *Self-Switching Weight* and *Correlated-Switching Weight* for identifying modules that are highly likely to cause large dI/dt problems.
- A force-directed floorplanner that considers module-level current-demand information and microarchitectural switching correlations to guide placement.
- A simulated-annealing-based floorplanning algorithm that incorporates microarchitectural feedback for module placement. This option is compared to the force-directed algorithm.
- A SPICE model of an on-chip power-delivery network that is used to evaluate the effectiveness of our approach. Based on the model, we present the maximal voltage swing at each module and the overall noise tolerance of the chip.

Our floorplanner allows the design of a floorplan that is inherently noise tolerant. However, it still cannot guarantee reliable operation during the worst-case noise scenario. To prevent the worst-case scenario, we also introduce a low-cost dI/dt controlling mechanism embedded into the microarchitecture that will dynamically limit high-frequency current-demand swings. This design can be integrated into the microarchitecture during the early

planning stages to facilitate the design of a processor for the average-case current-consumption scenario. This dynamic dI/dt controller is the second component of our design methodology and features the following:

- Decay counters used as a simple mechanism to monitor the access pattern of each microarchitectural module to prevent unsteady self-switching activity.
- A novel microarchitectural technique using a *queue-based dynamic dI/dt controller* to prevent simultaneous (or correlated) gating of modules that share the same local power-pins on the power delivery network.
- *Preemptive ALU Gating* that is integrated into our queue-based dynamic dI/dt controller to avoid performance loss and further reduce high-frequency dI/dt noise.
- An enhancement that performs progressive clock gating to achieve fine-grained dI/dt control for large modules without violating the current-demand threshold.

Unlike prior techniques [42, 48, 93, 52, 49] that largely aim to provide chip-level dI/dt control, our technique monitors and controls dI/dt by leveraging spatial information of modules obtained from a given floorplan and its power-pin distribution. This is the primary reason that our floorplanning algorithm and dynamic controller compliment each other perfectly. Inductive noise is highly dependent on the chip floorplan, which determines the relative location of functional modules and their distance from the power-pins. Hence, a solution at the chip-level is too coarse-grained and cannot account for the fact that power-pins are relatively unaffected by distant modules. For the same reason, such designs are also likely to generate many false alarms, resulting in undesired performance degradation. In contrast, by preventing simultaneous gating of modules that share the same power-pins, our proposed technique can accurately limit the current-demand swings to be within designated bounds.

The work presented in this chapter is a collaborative effort among several authors, including the author of this dissertation. Major portions of work from other authors are

included in this dissertation as necessary background material for understanding my specific contributions. Section 3.2, published as Mohamood *et al.* [94], contains a description of the dynamic noise controller architecture and its implications for microarchitecture design. The simulation infrastructure required to collect the self- and correlated switching activity metrics was developed by the first author of Mohamood *et al.* [94]. My major contributions to this work are the two floorplanning algorithms and related results described by Sections 3.3 and 3.4, published as Mohamood *et al.* [95] and Healy *et al.* [96], respectively.

3.1 Unified Design Methodology

3.1.1 Design Flow

An overview of this work's design flow is shown in Figure 22. The input to the flow is an architectural description and a set of benchmark programs. The size of each module in the floorplan is estimated using GENESYS [77] and eCACTI [82]. The flow begins with cycle-level microarchitectural simulation using SimpleScalar [78] and integrated power

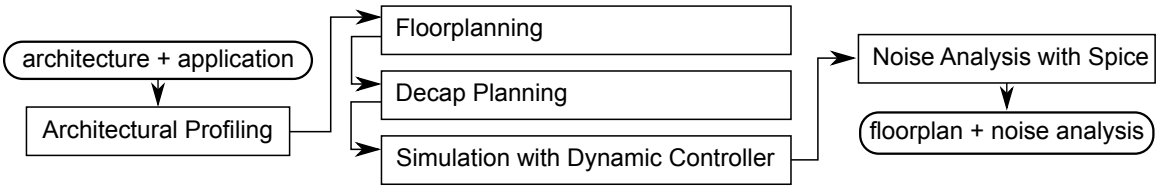


Figure 22. The design flow used in this work.

consumption estimation using Wattch [79]. During this simulation the power consumption of each microarchitectural block, and its switching activity, are collected on a per-cycle basis. This collection is done without the use of the dynamic noise controller. The switching activity factors are then used to optimize the floorplan. A large number of candidate floorplans are generated and stored during the floorplanning phase. Finally, decap planning is used to select the best among the candidate floorplans. This best floorplan is used to report our results.

3.1.2 Architectural Profiling

Architectural profiling is done using SimpleScalar, a cycle-level microarchitectural simulator. This work assumes aggressive and coarse-grained (module by module) clock gating on a cycle-by-cycle basis. Two statistics are collected about each module. These statistics are the self-switching weight, and the correlation weight. The self-switching weight is a normalized measure of how often a block changes power state and the correlated-switching weight is a normalized measure of how often a pair of blocks switch power states in the same direction during the same cycle. Highly correlated blocks are likely to cause power-noise problems and so should be placed far from each other in a noise optimized floorplan.

3.1.3 Floorplanning

This work compares two floorplanning algorithms. The first uses a force-directed method and is called Noise-Direct. The second method is based on simulated annealing with the Sequence Pair [8] floorplan representation. Floorplanning can impact noise problems in a power-distribution grid by moving noisier blocks both away from each other and closer to the power-pins. A longer current delivering path between a power-pin and a noisy block will lead to more noise seen by that block's neighbors and by itself due to the increased inductance in the longer path to the power-pin. Additionally, if the architecture utilizes a dynamic controller that is floorplan aware, as in this work, then it is possible to optimize the operation of that controller by providing it with a well formed floorplan. This work uses a new annealing cost function that specifically targets two sources of noise, as well as the physical basis of the dynamic noise control algorithm. More details for the cost function are described in Section 3.4.1. We also include new forces in our force-directed floorplanner that deal with power-supply noise. More details of the force-directed formulation are described in Section 3.3

3.1.4 Decoupling Capacitor Planning

Large amounts of decoupling capacitors (decaps) are used to reduce inductive noise in modern designs. This work uses a network-flow-based approach [97] to perform decap planning. The planning algorithm is used to select the best among a large group of low-cost (according to the cost function) floorplans found during annealing. The network flow algorithm is used to analyze how much decap each of the floorplans requires. The floorplan with the smallest requirement is chosen as the overall best. All white space in the floorplan is occupied by decaps.

3.1.5 Run-Time Noise Controller

Clock gating at the microarchitectural level is extensively used to control the total dynamic power dissipation of modern processors. In this work a dynamic noise controller based on a floorplan-aware set of queues is implemented to address the noise problems created by clock gating. The modules are selected to be placed in each queue of the controller based on their location in the floorplan. Modules that are close to one another are likely to cause noise problems for each other. The controller prevents modules within the same queue from switching in the same cycle, and each module's request to power off is limited by a decay counter. The decay counter prevents modules that are used regularly but not constantly from rapidly switching on and off in a short time frame. The worst-case noise scenario is when all modules attempt to switch simultaneously from one power state to another. The dynamic controller is designed to eliminate occurrences of this noise-inducing behavior, as well as other, less severe, events. Details of the run-time noise controller are provided in the next section.

3.2 Queue-Based Dynamic dI/dt Controller

The dynamic dI/dt controller is intended to address inductive noise issues resulting from excessive clock gating by improving the current-demand profile regardless of program behavior. Our design is easily customizable to enable a given design to achieve the right balance among dynamic dI/dt control, power consumption, and performance overhead. The primary components of the dI/dt controller are as follows:

- A low-overhead modular decay-counter-based clock-gating mechanism. The objective of the decay counters is to throttle excessive self-gating activity of modules.
- A floorplan-aware clock-gating queue that selectively disables simultaneous switching of modules in the same direction. The queue-based controller is designed to limit the maximum current surge or dip for a given set of power-pins shared by several modules on the power-supply grid.
- Preemptive activation of ALUs through pre-decoding for simultaneous dI/dt and performance enhancement.
- An enhancement to the queue controller that enables progressive clock gating of large modules, such as L2 cache banks.

3.2.1 Decay-Counter-Based Clock Gating

The key to avoiding clock-gating-induced noise lies in identifying program phases to determine whether clock-gating activity will impact power-supply reliability. Although certain elaborate techniques can accurately predict module requirement patterns, clock gating requires low-overhead mechanisms to justify the extra hardware cost. To enable a low-overhead dynamic clock-gating scheme that provides a tunable form of dI/dt control, we propose the use of decay counters. We can choose to save power only during long stretches of inactivity by using low-resolution decay counters to monitor module access patterns.

To illustrate this, Figure 23 provides an example that quantifies fine-grained module

access patterns for select modules over a small simulation period. The figure shows an example of access pattern profile for the branch predictor, the L1 I-Cache, an Integer ALU and the Integer Register File for the 256.bzip benchmark. The 200-cycle interval is shown here to illustrate the potential high-frequency dI/dt effects from a fine-grained perspective.

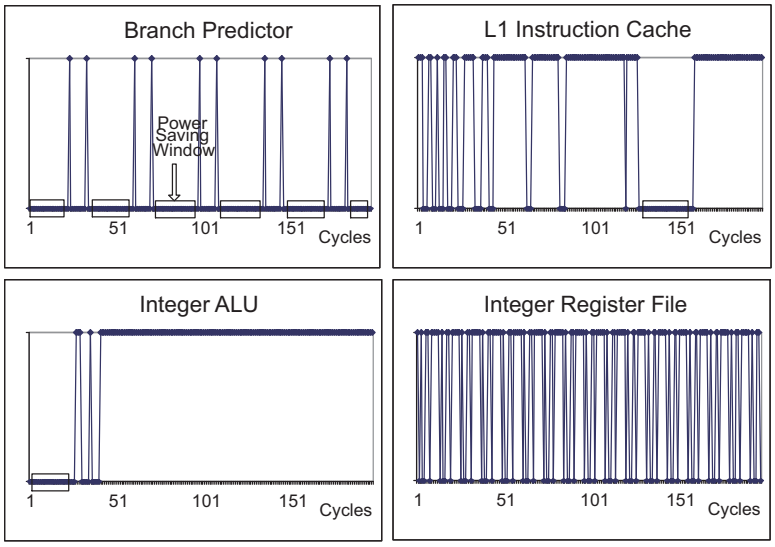


Figure 23. Module access patterns.

Typically, it is observed that a module that is inactive for more than 10-12 cycles is likely to remain dormant for an extended period of time. Clearly, there is a threshold cycle count beyond which a module can be gated off reliably with the least likelihood of encountering high-frequency inductive noise. On the other hand, it can be seen that when a module remains unaccessed for less than 5-10 cycles, it is highly likely to be accessed again soon.

Decay counters exploit this behavior by only enabling clock-gating activity when a minimum turn-off-time threshold has been exceeded. We use a 4-bit decay counter for each microarchitectural module inside the processor. The decay counters only permit clock gating of modules that have not been accessed during the last 16 cycles. For any given module, the counter decays every cycle unless there is an access made to that particular module, in which case the counter is reset back to the maximum.

The resolution of the decay counter provides the trade-off between high-frequency inductive-noise control and power dissipation. A large decay counter will further smooth

out current spikes over time, but at a cost of higher average power consumption due to the fact that modules will be gated off only after a long interval of inactivity. The opportunity for power saving is also dependent on the module access pattern. As shown in Figure 23, certain modules, such as the branch predictor or I-ALU, exhibit larger potential for power savings than others that display high activity, like the Integer Register File.

3.2.2 Dynamic dI/dt Controller Architecture

Though the decay counters can provide a smoother current profile for each module by eliminating unwanted switching activity, they are inherently incapable of avoiding dI/dt issues caused by simultaneous gating of modules that share common power pins. To address these shortcomings, we propose a *queue-based controller* that is aware of the processor's floorplan and power-pin distribution. In the processor's power-delivery network, a module usually draws more current from spatially nearby power pins, in other words, following the path(s) with the lowest impedance. Consequently, adjacent modules, if they switch simultaneously in the same direction, will stress local power-pins. Therefore, to guarantee that the maximum current ramp is below a specified limit at any given time, it is necessary to dynamically alter simultaneous gating of modules that are likely to stress the same power-pin(s). The proposed queue-based controller is designed to overcome reliability problems caused by simultaneous switching of adjacent modules. The salient features of the controller are as follows:

- A static queue with an entry for each module sharing the same power-pin domain. Ideally, there will be no more than eight entries in a queue resulting in a 3-bit module identification number that is local to each queue.

- Every queue entry contains the corresponding *state* of the matching module, which indicates either the current state or any requested clock-gating transition event. This will require 2 bits for the ON/OFF states as well as the ON→OFF and OFF→ON

transitions. The *state* is used to drive the pre-wired clock-gating signals to the corresponding modules.

- Every queue entry that represents a module also has an associated *integer weight* that is proportional to the current consumed by the corresponding module. We use a two bit integer to represent one of the four different current-consumption levels. Since weights are use to compute and check for current-demand violations, integer weights are appropriate for faster current-demand calculations. Fast calculations are essential for quick response to high-frequency dI/dt events.

A simplified version of our high frequency dI/dt controller architecture is depicted in Figure 24. The "+" signs on the chip floorplan (left-hand side) indicate the power-pin loca-

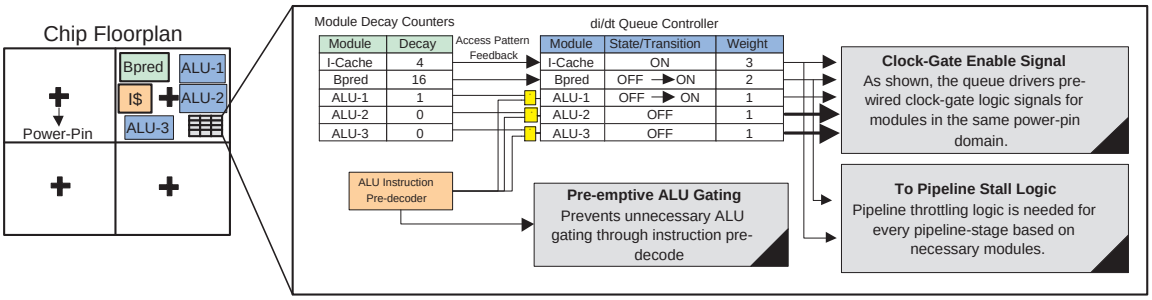


Figure 24. Noise controller architecture.

tions. For simplicity, we illustrate only four power-pins. The queue-based controller works in the following manner. The decay counters will signal a transition event, i.e. ON→OFF, for a given module in the queue. Let Δ be the current-demand threshold that is permitted for a given power-pin domain. At any given time, a *head* pointer is always pointed to one single module in the queue. Every cycle, the queue is traversed by a window size which has a total weight of Γ. The value of Γ is the largest sum of weights of consecutive modules that are in the transition states (ON→OFF or OFF→ON), such that $\Gamma \leq \Delta$. Since integer weights can be negative as well,[1] the sliding window will attempt to permit the maximum allowed transitions without violating the maximum current-demand constraint.

[1] OFF→ON is a positive switch while ON→OFF represents a negative switch.

To better understand the dI/dt queue-controller mechanism, we use an example based on the instantaneous state of the controller as shown in Figure 24. Let us assume that the value of the current-demand threshold is $\Delta = 3$. In the figure, ALU-2 and ALU-3 are gated off (indicated by the bold arrow that is the output of the queue controller). Both the Bpred and ALU-1 have an activation request indicated by the OFF→ON state. Therefore, the combined weight of the sliding window is $\Gamma = 3$.[2] The queue controller will therefore permit both module-gating events to occur, since the threshold constraint is not violated in this case. After servicing the transition, the *head* pointer will traverse two entries and point to the ALU-2 entry in the queue. In contrast, consider an alternate case where ALU-1 had a higher weight that resulted in the weight of the sliding window exceeding the current-threshold budget. In this case, only the Bpred transition will be serviced by the queue controller. Also, the *head* pointer will traverse only one module entry to ALU-1, so that it can be serviced in the next cycle. Furthermore, consider yet another example where ALU-1 requires an ON→OFF transition, which represents a negative weight. In this case, Γ=1, thus still permitting both Bpred and ALU-1 to perform their transitions. However, the sliding window value will still be below the threshold, Δ, and the queue-controller can potentially gate the next module, ALU-2, if it requires a transition. These examples are provided to illustrate how the sliding window adjusts dynamically based on the worst-case current demand that can be sustained by a given power-pin domain.

The example dI/dt queue in Figure 24 shows the modules in descending order of weight. It should be noted that the dI/dt controller will enforce the current demand threshold regardless of the order of the modules in the queue. However, the ordering of modules does affect the performance overhead imposed by the design. For instance, clustering modules in the queue that have high weights will create a larger performance overhead since multiple modules will not be permitted to transition simultaneously. The ordering of modules in the queue is static and presents a design choice that needs to be made by an architect for a

[2]Please note that in a real implementation, the sliding window will have an upper limit in terms of the number of module weights that can be computed together in a given cycle.

given floorplan.

Also, note that the queue in our dI/dt controller is different from a typical queue structure like the Instruction Fetch Queue, a memory structure allocated at run-time. In contrast, the entries in the dI/dt controller queue are pre-wired for each module at design time to simplify the logic for driving clock-gating signals directly to the modules. Functionality-wise, the controller behaves like a circular queue that traverses a number of modules determined by the sliding window threshold. It should be noted that the maximum hardware overhead of each microarchitectural module is merely 11 bits (including the decay counter). This amount is rather negligible in terms of additional power dissipated and extra current drawn by the controller itself.

3.2.3 Preemptive ALU Gating

We also propose using preemptive ALU clock gating through pre-decoding of instructions to prevent unnecessary gating activity. Decay-counter-based clock gating allows gating events to occur based on the history of module accesses. However, decay counters by themselves will be unable to predict the future switching activity of modules. For instance, it will be detrimental to performance if an ALU is going to be gated off due to a saturated decay-counter, when in fact an incoming ALU instruction has just been fetched. Furthermore, if an ALU instruction is on its way, it makes sense to leave the unit in the ON state even from a dI/dt perspective. To achieve this goal, we include preemptive *turn-on* gating of ALU modules by pre-decoding instructions. In a typical RISC ISA, the opcode can be determined by observing the first few bits of the instruction,[3] allowing us to pre-decode this information simultaneously with the instruction fetch. In the case that an ALU instruction has been detected early on, it is used to override the decay-counter *turn-off* request. In CISC ISAs, it might not be easy to perform a simple pre-decode due to variable length instructions. However, even in this case, other techniques, such as storing pre-decode information in the L1 Instruction Cache [98], can be used to achieve this effect.

[3]For example, the Alpha and PowerPC ISA uses the prefix 6 bits for the opcode.

3.2.4 Enhanced Progressive Gating of Large Modules

Simultaneous power-state switching of multiple modules can be prevented completely by selective gating. However, some monolithic modules like the L2 Cache can still consume large amounts of current, resulting in unreliable voltage swings. For this reason, certain processors employ progressive gating of large modules, like the L2 Cache, to mitigate dI/dt effects [91]. However, *ad hoc* progressive gating does not prevent other adjacent modules from switching simultaneously and can still result in unreliable dI/dt surges. To counteract this issue, our queue-based controller can be used to generate multiple clock-gating domains for even a single monolithic module by merely replicating multiple entries for a module with smaller weights. For instance, for a banked L2 cache, there can be as many entries as the number of banks within the queue with proportionally lower weights.[4] Since the queue inherently throttles simultaneous switching activity, it presents a much more effective progressive gating mechanism than current solutions. Thus, the queue-based controller can enable efficient progressive gating of such modules while maintaining the current demand below the noise-tolerant threshold, which is set to mitigate negative simultaneous-switching effects.

3.2.5 Pipeline Design Implications

The use of any dynamic dI/dt controller requires an appropriate performance throttling mechanism to guarantee program correctness even if certain necessary processor components are unavailable when needed. For instance, the instruction scheduler needs to be accurately aware of ALU availability before issuing operations. The integration of a dI/dt controller into a conventional architecture will require the pipeline logic to be fully aware of the clock-gating state of the modules to issue operations without affecting correctness. For this reason, it is essential that the dI/dt controller not impose impractical design implications on the processor pipeline.

Our queue-based high-frequency dI/dt controller can be easily built into a conventional

[4]Typically, L2 cache banks are in separate clock-gating domains.

out-of-order pipeline without significant additional complexity. Conventional processor modules are already capable of correctly operating under resource contention. In the event of resource hazards such as over-subscription of ports in the register file, caches, or load-store queue, the selection logic will properly delay the issue of certain operations. As indicated in Figure 24, our queue has static entries and pre-wired logic that indicates the availability of any given module. This allows efficient integration of the additional resource-availability constraints into the existing selection logic in the pipeline. Since resource availability can be directly interpreted from the output of the queue-based controller, an enhanced pipeline with the dI/dt controller merely needs to ensure that the resource-availability constraint overrides all conventional hazards to ensure correct functionality.

3.3 Noise-Direct Floorplanning Algorithm

Noise-Direct is a profile-driven force-directed floorplanning technique. The noise experienced by a module or a power-pin is dependent on the activity of the microprocessor and the application running on it. Information about the microarchitectural activity is used to guide the placement of modules across the microprocessor power grid. Therefore, a microarchitectural simulator is used to quantify module activity as a function of a given set of benchmarks. Once the average behavior of a given module with respect to all other modules is obtained, a noise-aware floorplan can be created with any appropriate floorplanning algorithm that leverages this information. The next section describes the method used to obtain the switching activity of various modules and how this information is utilized in the Noise-Direct floorplanner.

3.3.1 Quantifying Module Activity

To effectively manage and prevent high-frequency voltage ringing at the microarchitectural level, it is important to understand the simultaneous-switching behavior among different microarchitectural blocks and their relative locations to each other (i.e. the sharing of power-pins) on a given floorplan. Two metrics are used to understand the switching

behavior of microprocessor modules: the self-switching activity and the correlated- or simultaneous-switching activity of modules. The self-switching measurement is used to quantify the number of gating occurrences of a given module during the profiling period. Both gating ON and OFF are considered events that are likely to cause current fluctuation. The objective of this metric is to single out microarchitectural modules that exhibit high switching activity. In addition, the intensity of the gating activity also depends on the current consumption of each module. In other words, even if a module switches less frequently than the others, it still can induce intolerable noise if it draws a significant amount of current. The relative number of switching events and the current consumption per cycle, called *intensity of switch*, are combined into a single weight. If sw_i represents the relative number of switching events for module i and I_i is the intensity of the switch, then the self-switching factor, α_i, is represented by the following relationship:

$$\alpha_i = sw_i \times I_i. \tag{1}$$

Correlated switching events are gating events in the same direction i.e. both modules are clock gated ON or OFF simultaneously. The inter-cycle gating direction of each module is captured in the profiling process to measure the correlation. Then, each module is paired with every other module in the processor and checked for simultaneous gating in the same direction. The result is an upper triangular correlation matrix with each location representing the number of simultaneous gating events encountered. An illustration of the process used to calculate correlated-switching events is given as follows:

$$\begin{bmatrix} \alpha_1 & \frac{1}{2}\left(\frac{X_{12}}{sw_1} + \frac{X_{21}}{sw_2}\right) & \dots & \frac{1}{2}\left(\frac{X_{1n}}{sw_1} + \frac{X_{n1}}{sw_n}\right) \\ 0 & \alpha_2 & \dots & \frac{1}{2}\left(\frac{X_{2n}}{sw_2} + \frac{X_{n2}}{sw_{n-1}}\right) \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & \alpha_n \end{bmatrix}$$

In the matrix, X_{ij} is the number of raw correlated switches that occurred over the profiling duration and sw_i is the number of self-switching events for module i. Note that the

correlation metric X_{ij} isolates only the modules in consideration. The upper bound of 100 indicates a perfect correlation, i.e. modules *i* and *j* switched simultaneously every single time[5] (100% of the switching events). The forward diagonal in the matrix represents the self-switching factor, α_i, for each module.

Using eight SPEC2000 INT benchmark programs,[6] Table 5 shows the switching correlation as a matrix for the 23 microarchitectural modules considered in the processor model.

	LSQ	RUU	BTB	L2$	IRF	L1D$	ALU0	ALU1	ALU2	ALU3	ALU4	ALU5	L1I$	Bpred	DTLB	ITLB	FALU0	FALU1	Freg
LSQ	*28*	0	20	13	20	2	10	10	10	10	10	10	11	20	0	11	10	10	12
RUU		*26*	8	4	13	2	0	0	0	0	0	0	5	8	2	5	0	0	5
BTB			*18*	7	29	17	13	13	13	13	13	13	37	100	17	37	13	13	13
L2$				*16*	14	28	12	12	12	12	12	12	21	7	26	21	4	4	7
IRF					*10*	17	7	7	7	7	7	7	23	29	17	23	8	8	24
L1D$						*7*	6	6	6	6	6	6	11	17	93	11	5	5	6
ALU0							*3*	100	100	100	100	100	15	13	6	15	66	66	4
ALU1								*3*	100	100	100	100	15	13	6	15	66	66	4
ALU2									*3*	100	100	100	15	13	6	15	66	66	4
ALU3										*3*	100	100	15	13	6	15	66	66	4
ALU4											*3*	100	15	13	6	15	66	66	4
ALU5												*3*	15	13	6	15	66	66	4
L1I$													*3*	37	12	100	11	11	5
Bpred														*3*	17	37	13	13	13
DTLB															*2*	12	5	5	6
ITLB																*1*	11	11	5
FALU0																	*1*	100	5
FALU1																		*1*	5
Freg																			*0*

Table 5. Self- and correlated-switching weights of modules for a sample benchmark.

The diagonal (gray) in the matrix represents the amount of *self-switching* factor. As can be observed from Table 5, certain modules switch far more frequently than others. On the other hand, the weights of the modules that are likely to be accessed every cycle (turned ON most of the time), such as the L1 I-Cache and the I-TLB, are lower. Some modules with smaller weights are dormant. For example, the floating-point register file (Freg)[7] is accessed very infrequently. As expected, the branch predictor and the BTB, and the I-Cache and the I-TLB (and the D-Cache and the D-TLB) are all highly correlated modules. In addition, it is also observed that the first six ALU modules are also highly correlated because concurrency exists in integer instructions. The design of the high-frequency dynamic dI/dt

[5]Note that the correlated switching factor was scaled as a percentage, hence the upper bound is 100.

[6]Since correlation profiling was compute-intensive, we used the following subset of the SPEC2000 benchmarks for this motivational data: 256.bzip, 186.crafty, 252.eon, 254.gap, 164.gzip, 181.mcf, 253.perl and 300.twolf.

[7]Table 5 is a demonstration and contains only results for the integer benchmark programs.

controller is mainly based around the intrinsic switching behavior of microarchitectural modules.

The techniques presented in this chapter address high-frequency inductive-noise issues directly caused by clock gating. Clock gating is a well-established method of dealing with increasing power demand. The main reliability issue associated with clock gating is that there is no deterministic or predictive way for determining whether it is acceptable to gate off modules without inducing hazardous current surges. In addition, conventional clock-gating methods do not take into account any knowledge of adjacent modules and the extent of correlated clock-gating activity. Our cooperative floorplanning and microarchitectural level high-frequency dI/dt control is based on fundamental observations of the clock-gating activity of modules: correlation with adjacent modules, the module locations in a floorplan, and the power-pin distribution of the chip.

3.3.2 Overview of the Floorplanner

Given a set of microarchitectural modules and a netlist that specifies the connectivity among these modules, the Noise-Direct microarchitectural floorplanner tries to determine the location of the modules in a chip such that (i) there is no overlap among modules, (ii) the sum of current demand for each power pin does not exceed its capacity, and (iii) power-supply noise experienced by each module does not exceed the given bound. The objective is to provide a floorplan that minimizes area and total wirelength. Microarchitectural floorplanning has drawn significant interests from both the computer architecture and EDA communities [14, 12, 13, 15, 99]. These existing works mainly target performance and thermal issues. Power-supply noise has not been addressed by previous floorplanners.

Among several methods known for floorplan optimization, the force-directed floorplanning method [100] has been chosen. Compared with other methods, such as simulated annealing [8], the slicing method [4], and analytical approaches [7], the force-directed method require less tedious parameter tuning and converges quickly while obtaining high-quality solutions [100]. The force-directed floorplanning problem is formulated as finding a set of

forces. These forces exist among and between fixed objects (such as I/O or power pins) and movable modules, and optimize the objective function. The problem of finding module position then becomes one of finding forces. The floorplanner consists of the following four steps:

1. Initialization: To begin, all modules are randomly distributed throughout the placement area, without regard to overlap.

2. Iteration: The objective function is optimized in an iterative manner, where a certain set of forces is updated based on the previous iteration to guide the optimization process.

3. Stopping Criterion: The iterations are stopped when the utilization of the floorplanning area is above a threshold. This has the effect of an overlap constraint as the floorplan area is related to the sum of the area of the blocks and the utilization cannot go above a certain level without a corresponding drop in the amount of overlap.

4. Legalization: The final legalization step removes the overlap among modules while maintaining the high quality of the force-directed solution.

The objective function contains the following types of forces (see Figure 25 for reference). (1) Net force (F_{net}): all pins in the same net are pulled closer together to minimize the wirelength objective. (2) Center force (F_{cen}): all modules are pulled to the center of the chip to discourage the modules from moving beyond the chip boundary. (3) Correlation force (F_{cor}): modules with high switching activity repel each other so that the noise caused by the modules is reduced. The correlation factors $\gamma_{i,j}$ described in Subsection 3.3.1 are used to compute the magnitude of the correlation force. (4) Density force (F_{den}): modules located in a high density region of the chip are pushed apart to reduce overlap. (5) Pin capacity force (F_{pin}): modules are pulled towards or pushed away from each power pin so that the total demand on each power pin is evenly distributed and its capacity is not

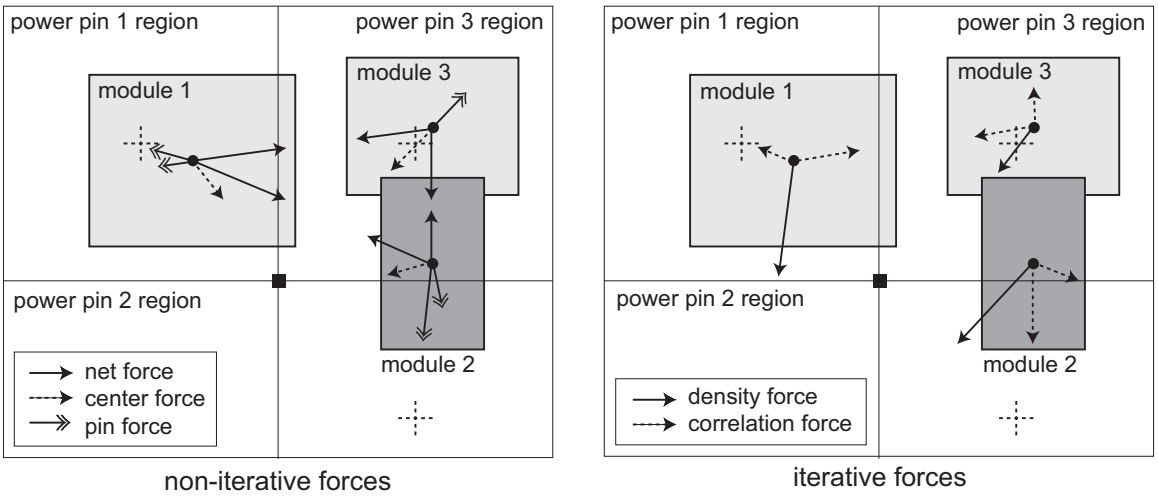


Figure 25. Illustration of various forces optimized by the Noise-Direct floorplanner.

violated. The first three types are non-iterative, whereas the last two are iterative. All the non-iterative forces are fixed during the floorplan optimization process, whereas the iterative forces are updated based on the previous iterations. The following combined force is optimized to balance the impact of the five types of forces:

$$F_{tot} = \lambda \cdot F_{net} + \theta \cdot F_{cen} + \mu \cdot F_{cor} + K \cdot F_{den} + \rho \cdot F_{pin}, \tag{2}$$

where λ, θ, μ, K, and ρ are weighting constants.[8]

3.3.3 Force Equations

Let n be the number of free modules in the floorplan and (x_i, y_i) be the x and y-coordinates of the center of module i, respectively. A placement can be described by the $2n$-dimensional vector $\vec{p} = (x_1, \ldots, x_n, y_1, \ldots, y_n)^T$. The cost of a connection is then formulated such that it is proportional to the squared Euclidean distance between its endpoints. The objective function sums the cost of all connections and therefore can be written in matrix notation as:

$$\frac{1}{2}\vec{p}^T C \vec{p} + \vec{d}^T \vec{p} + const, \tag{3}$$

[8]The empirically derived choice of these values is to set them all equal. These weights are fixed to be constant during the entire floorplan optimization process. One can tune the weights statically or dynamically to emphasize desired objectives.

where the $2n \times 2n$ symmetric matrix C and the vector $\vec{d}$ are produced from the module connections and their weights, and the formula for squared Euclidean distance. For example, the x-part of the connection between two free modules i and j is $(x_i - x_j)^2 = x_i^2 - 2x_i x_j + x_j^2$. The first term adds to $C_{i,i}$, the second term to $C_{i,j}$ and $C_{j,i}$, and the third term to $C_{j,j}$. Similarly, for a fixed connection between free module i and fixed location f, $(x_i - x_f)^2 = x_i^2 - 2x_i x_f + x_f^2$ adds the first term to $C_{i,i}$, the second term to $\vec{d}_i$, and the third term to the constant part of Equation (3). This cost function is minimized by solving the linear system:

$$C\vec{p} + \vec{d} = 0. \tag{4}$$

This formulation is equivalent to modeling connections as springs and calculating the state of equilibrium.

Force-directed floorplanning and placement algorithms are well known for their overlap problems. Spreading or repulsive forces are required to make the final solution feasible, i.e. with zero overlap. These additional forces extend Equation (4) with the force vector $\vec{e}$ to model constant additional forces that are iteratively updated:

$$C\vec{p} + \vec{d} + \vec{e} = 0. \tag{5}$$

The complexity of solving this equation is $O(k \cdot n^2)$, where k is the number of iterations, and n is the number of modules. Experimentation shows that k ranges from one to ten and n is around twenty. Thus, this algorithm generates optimized solutions quickly.

The pin capacity force is computed as follows. Let the "current drawing region" of a pin be defined as a rectangle centered on that pin with width and height equal to the distance between pins. Let c_i be the power consumption of module i located within the current drawing region of power pin j, I_j be the capacity of power pin j, (x_i, y_i) be the center of module i, (x_j, y_j) be the location of pin j, and $d_{i,j}$ be the squared Euclidean distance between module i and pin j. Let α_i be the self-switching weight of module i as defined in Subsection 3.3.1. The x direction force between free module i and fixed pin j is then:

$$F_{pin}^{x}(i, j) = \left[\left(\frac{I_j}{\sum_i c_i}\right)^2 - 1\right] \cdot \frac{|x_i - x_j|}{d_{i,j}} \cdot \alpha_i. \tag{6}$$

A similar definition follows for the force along the *y* direction. This force is proportional to the distance between the module and the pin, negative if the sum of the current being drawn from the modules in the current drawing region of the pin are greater than the capacity of the pin and positive otherwise, and in the range $(-1, \infty)$. Basically if the demand of block i is higher than the capacity of the pin j, then the force pushes the modules away; otherwise, it pulls the modules towards the pin.

3.3.4 Updating Iterative Forces

As mentioned previously, two kinds of forces are updated during each iteration: F_{den} and F_{pin}. Specifically, the location of the modules from the previous iteration is used to recompute the density of each region in the floorplan as well as the attractive or repulsive forces among the modules within the vicinity of each power pin. The main motivation for this force update is to satisfy the non-overlap constraint (via updating F_{den}) and pin capacity constraint (via updating F_{pin}). In case these constraints are not met in the current solution, the amount of violation is minimized as much as possible by attempting another iteration. Note that the pin constraint is easily satisfied, but not the overlap constraint. Thus, the post-process explicitly removes the overlap among the modules. Since $\vec{e}$ consists of F_{den} and F_{pin}, $\vec{e}$ gets updated and solved in each iteration.

3.3.5 Legalization

A simple heuristic is used to legalize the floorplan of the modules. Vertical and horizontal constraint graphs similar to those used for the Sequence Pair [8] floorplan representation are created based on the force-directed floorplan solution. The basic idea is to derive the relative positions among the modules based on the force-directed floorplanning, and then use Sequence Pair to encode them to remove overlap. For each pair of modules, the horizontal and vertical distance between their centers is compared. If the horizontal distance is smaller than the vertical distance then the appropriate constraint is added to the vertical constraint graph. Conversely, if the vertical distance is less, the appropriate constraint is

added to the horizontal constraint graph. If the modules overlap, then these constraints will push the modules apart in the direction that minimizes overall movement. Thus, the legalized modules remain close to their original locations. The constraint graphs ensure that the final floorplan is non-overlapping.

3.4 Noise-Controller-Aware Floorplanning Algorithm

This section introduces a simulated annealing-based queue-aware floorplanner. This floorplanner is compared to Noise-Direct in Section 3.5.

3.4.1 Annealing Cost Function

A new annealing cost function is used that specifically targets two factors affecting power-supply noise, as well as the physical basis of the dynamic noise controller. There are five terms in the cost function. The first two target traditional physical design objectives, area, A, and wire-length, W. The third term of the cost function, I, addresses self-induced inductance and IR drop. Correlated-switching factors are considered in the fourth term, C. And the final term, Q, includes consideration of the dynamic noise control algorithm. The total cost function is given by:

$$Cost = \alpha \cdot A + \beta \cdot W + \gamma \cdot I + \delta \cdot C + \epsilon \cdot Q, \tag{7}$$

where, α, β, γ, δ, and ϵ are weighting constants. In this work the values for the weighting constants were empirically determined to be 1, 0.2, 0.5, 0.5 and 0.025 for α, β, γ, δ, and ϵ, respectively. The first two terms are defined in the usual way based on Manhattan distance and the bounding box of the floorplan. The three final terms, and their use for controlling the three sources of noise and optimizing the performance of the dynamic noise controller, are described in the following subsections.

3.4.2 Self-Switching Current

The self-switching current term, I, is defined as follows:

$$I = \sum_{\forall i \in B, \forall j \in P} curr_i \cdot sw_i \cdot dist_{i,j} \cdot reg_{i,j}, \tag{8}$$

where B is the set of all blocks, P is the set of all pins, $curr_i$ is the current requirement of block i, sw_i is the self-switching factor of block i, $dist_{i,j}$ is the Euclidean distance between block i and pin j, and $reg_{i,j}$ is one if and only if block i is in the current drawing region of pin j, and zero otherwise. The current drawing region of a pin is defined to be half the distance to the next nearest pin. Figure 26 shows an illustration of the self-switching term. Previous work that considered the LdI/dt problem [95] did not directly consider the IR drop problem. The pin capacity force described in that work focused on satisfying the current

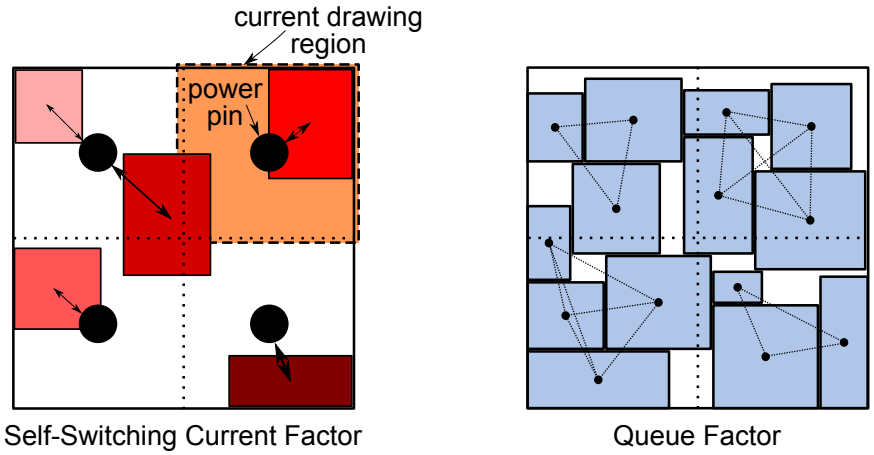


Figure 26. Illustration of the self-switching-current factor and queue factor cost function terms. For self-switching current, the higher-weighted (darker) blocks, based on current demand and switching activity, are drawn to the power-pins more strongly. For the queue factor, each quadrant of the chip has a different queue. Only modules within the same queue are given a weighted cost function bonus based on their correlated-switching activity and current demand. Queues are defined spatially and blocks have no movement restrictions during annealing.

drawing requirements of each pin, pushing blocks away from pins that were overloaded and pulling them towards pins that were underloaded. Their work did not weight blocks that needed more current than others. The self-switching term used in this work, I, considers both the current requirements of each block and the amount of switching that that block

exhibits. It therefore considers both the IR drop seen by each block as well as the self-induced inductance noise experienced by each block. When blocks are farther away from the pins, the resistance term, R, of IR drop is increased. In a complementary fashion, the inductance term, L, of LdI/dt inductance noise is increased when blocks are farther away from pins. The distance between pins and blocks that have high current demand and high switching activity is minimized by minimizing I. Therefore, the IR drop and LdI/dt noise seen by the chip as a whole is also minimized.

3.4.3 Correlated-Switching Factor

The correlated-switching factor, C, is defined as follows:

$$C = \sum_{\forall i,j \in B} \frac{curr_i \cdot curr_j \cdot corr_{i,j}}{dist_{i,j}}, \tag{9}$$

where $corr_{i,j}$ is the correlated-switching activity, described above, between blocks i and j. The rest of the terms are as defined in Section 3.4.2: B is the set of all blocks, $curr_i$ is the current drawn by block i, and $dist_{i,j}$ is the Euclidean distance between blocks i and j. The minimization of this term maximizes the distance between blocks that have both high current and often switch simultaneously. If two blocks are positioned near one another and draw current from the same power-pin, then simultaneous switching would exacerbate the LdI/dt noise seen by both blocks. An example table of the per-module correlated-switching weights for a sample benchmark is shown in Table 5.

3.4.4 Dynamic Controller Queue Factor

The dynamic controller queue factor, Q, is defined as follows:

$$Q = \sum_{\forall i,j \in B} -(curr_i \cdot curr_j \cdot corr_{i,j} \cdot q_{i,j}), \tag{10}$$

where $q_{i,j}$ is one if and only if blocks i and j reside within the same dynamic noise controller queue, and is zero otherwise. The rest of the terms are as defined above: B is the set of all blocks, $curr_i$ is the current drawn by block i, and $corr_{i,j}$ is the correlated-switching activity between blocks i and j. Figure 26 shows an illustration of the queue factor. The

dynamic noise controller is floorplan aware through the use of spatially organized queues. Therefore, by specifically optimizing the blocks that occupy each queue it is possible to optimize the operation of the dynamic noise controller through physical design. Including this type of optimization into a force-directed approach would be extremely difficult; that is why this work uses the more flexible simulated-annealing-based floorplanning approach. The queue factor takes its form from the correlated-switching factor. It adds consideration of current-weighted correlated-switching activity to the cost function based on whether two blocks share the same queue or not.

The initial motivation behind the form of the queue factor was that we believed it would be more problematic if highly-correlated blocks resided within the same queue. Highly-correlated blocks should be separated from one another with a large distance. If those blocks are within the same queue it will increase noise, because the queues are spatially designated and blocks in the same queue are necessarily close to one another. Experimentation proved this assumption to be incorrect. Floorplans produced using a cost function with a positive queue factor (+Q) had uniformly worse power-supply noise than floorplans produced using no queue factor (No Q) at all. However, if the queue factor is included as a bonus (-Q) instead of as a penalty, the noise characteristics are improved over a noise-aware-only floorplan. These results are discussed in more depth in Section 3.5.5. This behavior can be explained by the fact that no matter how far away highly-correlated blocks are from one another they are still connected to the same power distribution grid and can cause coupled inductive noise by switching simultaneously. It is also possible that highly-correlated blocks are near one another, but just slightly over the queue boundary. This scenario would cause noise problems but be given a lower cost with a positive queue factor. By adding a negative bonus (-Q) to the cost function when blocks that are highly correlated reside within the same queue, the dynamic controller is allowed to deal with the noise problem more effectively.

3.4.5 Power Network Analysis

A SPICE model of the on-chip power delivery network [45] is used to evaluate the effectiveness of the cost function that is used to guide the dynamic-noise-controller-aware floorplanners. A depiction of the SPICE model is shown in Figure 27. The noise-mitigation technique is evaluated using the worst-case current consumption scenario. The worst-case

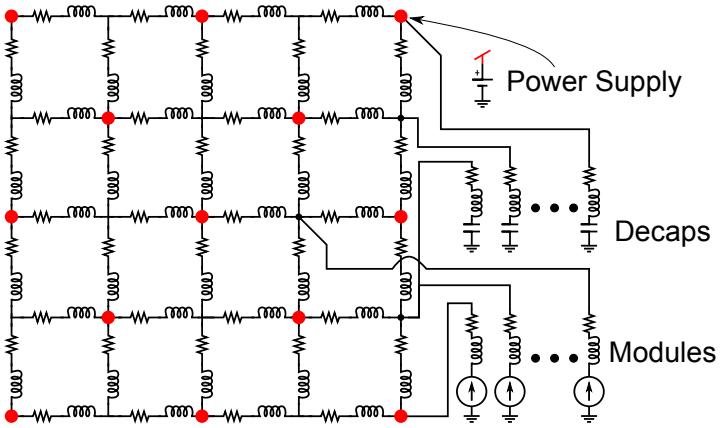


Figure 27. The SPICE circuit used for simulating LdI/dt noise. Voltage-supply bumps are positioned at every other grid point. Decoupling capacitors and modules (current sources) are connected to the nearest grid point in the floorplan.

switching activity of an application is determined by sampling the microarchitectural activity of all modules over the duration of the simulation. The period with the highest module-switching activity can be determined by comparing the activity during different program phases. Once the worst-case phase is identified, the current profile of each module is generated from the microarchitectural simulator. This complex current waveform is used as piece-wise linear (PWL) source input to the SPICE model. Induced noise effects can then be observed as a direct function of the application's behavior.

3.5 Experimental Results

This section describes the experimental techniques and simulation framework used to implement and evaluate the dynamic dI/dt control mechanism, the force-directed floorplanner, and the controller-aware floorplanning algorithm.

3.5.1 Simulation Framework

Our simulation framework is based on SimpleScalar 3.0 and Wattch [79] running the SPEC2000 INT and FP benchmark suite. To understand the access patterns of individual modules that motivated the solution of this work, we include various profiling and instrumentation facilities in our simulator. For the implementation of the dynamic dI/dt controller we extended SimpleScalar/Wattch to incorporate a floorplan-aware queue configuration. We also implemented a detailed, floorplan-dependent performance throttling model and queue configuration for studying the performance impact of our technique. The primary simulation parameters used in our simulations are shown in Table 6. The power and current consumption metrics were based on a 5GHz processor developed using a 70nm

Table 6. The microarchitecture's parameters.

Parameters	Values
Fetch/Decode width	8-wide
Issue/Commit width	8-wide
Branch predictor	Combining: 16K entry Metatable Bimodal: 16K entries 2-Level: 14 bit BHR, 16K entry PHT
BTB	4-way, 4096 sets
L1 I- and D-Cache	16KB 4-Way 64B line
I- and D-TLB	128 Entries
L2 Cache	256KB, 8-way, Unified, 64B line
L1/L2 Latency	1 cycle / 6 cycles
Main Memory Latency	500 cycles
LSQ Size	64 entries
RUU Size	256 entries
Functional Units	8 IntAlu (only 2 can be used for IntMult) 4 FPAlu (only 2 can be used for FPMult)

process technology. Each simulation was fast-forwarded by 4 billion instructions and simulated for 1 billion instructions. The current signatures that were chosen to evaluate the dynamic dI/dt controller represent the worst-case overall module switching activity over the entire simulation period.

3.5.2 Baseline Floorplanning Methodology

Our baseline floorplan is optimized with purely wirelength and area objectives. The baseline floorplan is independent of running applications, i.e. no profile-guided optimizations were employed during floorplanning. This baseline is compared against the noise-aware floorplan with the dynamic dI/dt control mechanism in place. The floorplan, along with the predefined power-pin distribution, determined the configuration of the queue entries used in the dynamic dI/dt controller.

3.5.3 Dynamic dI/dt Controller Results

The basic objective of our dynamic dI/dt controller is to minimize the burden on power-pin(s) caused by nearby modules. Therefore, for any given floorplan and power-pin configuration, the design objective of the dI/dt controller is to place queues for effective dI/dt control in a distinct section of the floorplan. For this work, we divided the floorplan into four quadrants, with each quadrant representing a distinct power-pin domain. Note that certain power-pins can be in multiple domains. For instance, quadrant-based module separation could result in five power-pins per quadrant, because the power-pins on the borders of the quadrants exist in multiple domains. The number of distinct power-pin domains is a design choice influenced by the degree of dI/dt control that is required. A high number of power-pin domains results in a larger number of queues and finer-grained control. On the other hand, too few power domains will result in larger queues impacting performance, because of the fact that the worst-case delay for transitions is higher. For our experiments, queues were assigned to each quadrant and all the modules placed in that quandrant. Since the floorplan determines the queue configuration, different floorplans will have different performance impacts as well as distinctive dI/dt characteristics.

We applied the technique to both the baseline floorplan and the controller-aware floorplan to evaluate the effectiveness and overhead of our dynamic dI/dt controller under different scenarios. The results include current profiles on a baseline machine without a dI/dt

controller versus our technique as well as the average current variability across all benchmarks. We also present the performance overhead incurred due to our dynamic dI/dt controller. Finally, we show the thermal impact of the controller, which is caused by the additional power dissipated by modules that would have been clock-gated in a design without a dynamic noise controller.

3.5.3.1 Current Profile of Applications

To demonstrate the effectiveness of our controller in improving high-frequency current-demand changes (dI/dt), we now present the current profile of the whole chip as well as the current profile for each queue cluster for the wirelength-driven floorplan. Note that the effectiveness of a dI/dt controller is evaluated by observing its effect on the worst-case current profile of a given application, which represents the computation phase with maximum module switching activity. Due to the staggeringly large number of current profiles for all of the benchmark programs, we demonstrate representative characteristics using two types of benchmark programs. Note that the crucial information conveyed in this section is to show the effectiveness of our proposed mechanism.

We profiled one high-ILP benchmark (164.gzip) and another low-ILP (181.mcf, memory-bound) benchmark. The current profiles shown in Figure 28 and Figure 29 were obtained by determining the worst-case switching activity during the course of execution. A 4-bit decay counter was used for each module in all experiments.[9] Each graph shows the current profile for both the processor with ideal clock gating as well as the decay-counter-based clock-gating mechanism. We also provide close-up versions of the representative, highly-active region of the graph for better visibility.

It can be seen that both 164.gzip and 181.mcf exhibit a repetitive current profile during the worst-case switching period. This is especially prominent in the current profile of 181.mcf where there is a period of high activity for a few hundred cycles, followed by a stable current profile for approximately 500 cycles. This is due to the long-familiar

[9]The resolution of the decay counter was based on the motivational data discussed in Section 3.2.2.

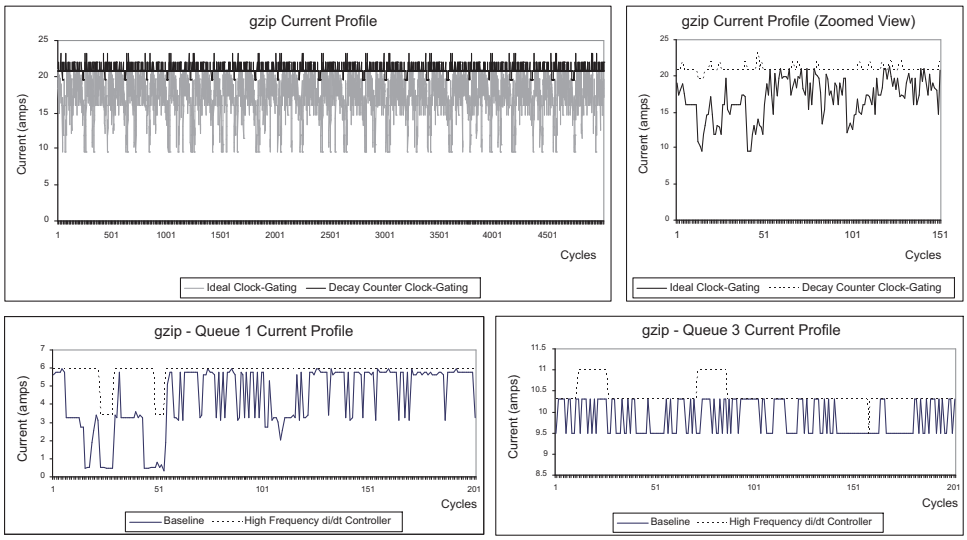


Figure 28. High ILP benchmark (164.gzip**) chip current and queue current profiles.**

cache misses to main memory that occur in 181.mcf. During this period most modules are inactive and can be clock-gated off to save dynamic power. The effectiveness of the dI/dt controller in improving the current ramp is obvious in the zoomed versions of the graphs. It shows that with the decay counter, our system (shown in dashed lines) successfully prevents unnecessary oscillating swings in the current profile and produces a much smoother down-ramp. For 164.gzip in Figure 28, we observe large current variation in the ideal-clock-gating scheme due to high activity across all modules because there is no significant duration of time where reasonable power savings are possible. Because of this, modules are never inactive for extended periods of time, and the decay counters rarely clock-gate off the modules. The current profile is extremely stable for this reason. In short, the decay-counter-based technique finds the optimal power envelope right above the ideal clock-gating mechanism and allows clock gating only when there is a significant chance that the given modules will not be accessed again soon.

Next, we present the current profile with the integration of the complete queue-based controller. Note that this is the complete controller that incorporates prevention of simultaneous switching, decay-counter-based feedback for clock gating, preemptive ALU gating and progressive gating of L2 cache banks. The lower boxes of Figures 28 and 29 also show

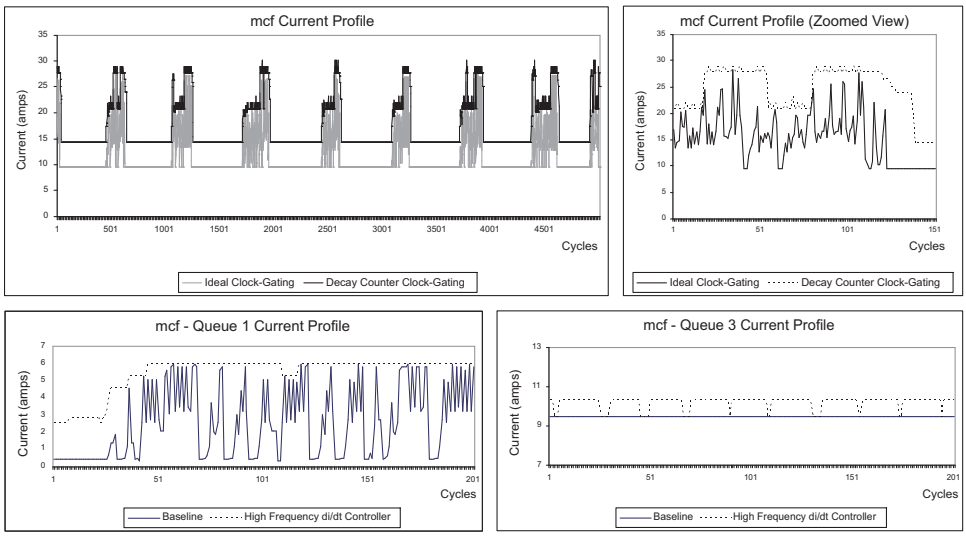


Figure 29. Low ILP benchmark (181.mcf**) chip current and queue current profiles.**

the current profile for Queues 1 and 3 for 164.gzip and 181.mcf. In all cases it can be observed that the current profile is significantly improved by eliminating excessive switching activity. In addition, both the upward ramp and downward ramp of the current demand are spread out across multiple cycles. This is more prominent in the upward ramp of the current with the dI/dt controller between cycle 20 and cycle 50 for *Queue 1* in 181.mcf. For *Queue 3* in 181.mcf we observe a different trend whereby the dI/dt controller ramps up current repeatedly compared to the baseline, which is stable. This is due to the preemptive ALU gating effect that ramps up additional ALUs that are otherwise unused in the baseline clock-gating scheme due to low ILP. We observe a repetitive pattern where ALUs are gated preemptively only to later decay after approximately 20-25 clock cycles. However, these ramps are still spread out over many cycles and do not violate the current demand threshold.

For 164.gzip, where there is high ILP/switching activity, the queue-based controller ramps up to the required current levels and does not saturate the decay counters for long enough. For this reason, the queue-current profile is almost always stable, except for the few cases where the decay counters decay long enough to enable clock gating. It is important to note that this does not mean that there is no opportunity for power-savings in

such a design without dI/dt control. The presented phase of 164.gzip is the highest ILP portion in our simulation and it is simply not worth it to clock-gate elements during this phase because of the dI/dt and performance penalties

Since presenting detailed current profiles is infeasible for all benchmarks, we now present the per-cycle current variability for the complete duration of the benchmark execution. Unlike the worst case profile that was presented earlier, this metric presents the *average per-cycle current variability* for both the baseline processor and the processor with our dynamic dI/dt controller. Figure 30 shows the comparison for various SPEC2000 INT and FP benchmark programs. The current variability is calculated by measuring inter-cycle

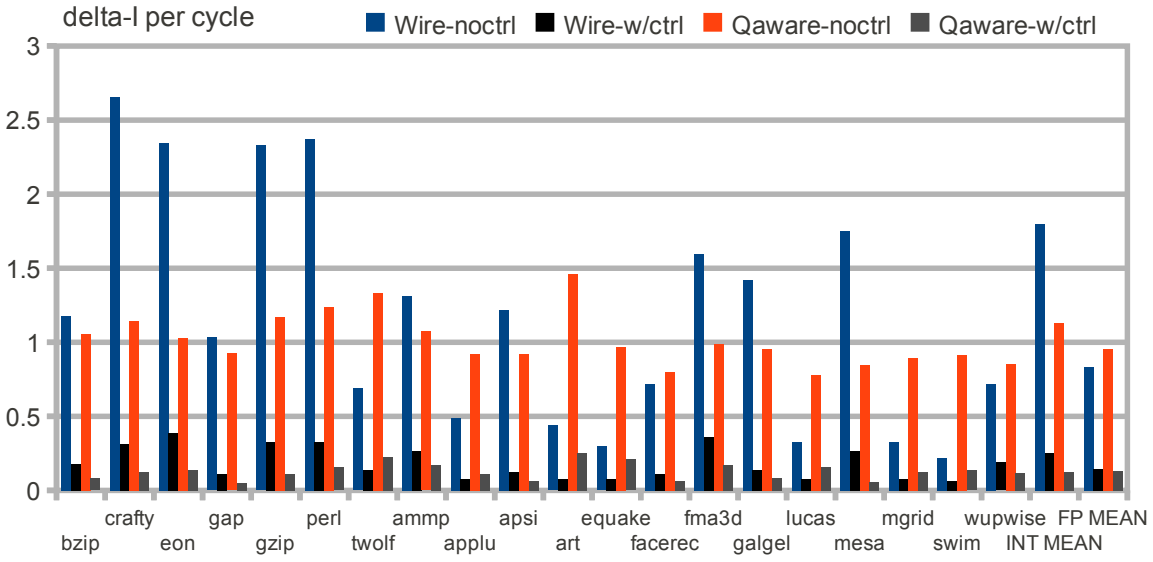


Figure 30. Average current variability for the benchmark suite with both the wirelength-aware and queue-aware floorplans.

current fluctuations (in absolute value of the swing) over the entire simulation period, as a fraction of the total number of simulation cycles. It can be observed that the baseline architecture shows a higher degree of current variability across the board. The data show that 186.crafty exhibits the highest variability whereas 171.swim has the lowest variability for the wirelength-driven floorplan. Regardless of the native current variability our dynamic dI/dt controlling mechanism can significantly mitigate the dynamic oscillating behavior of

the current profiles of running applications. The dI/dt controller pushes the current variability below 0.5 amps/cycle for all the benchmark programs studied.

3.5.3.2 Performance Impact

We now present the performance analysis of our dI/dt controlling mechanism. Figure 31 shows the IPC degradation for the SPEC2000 INT and FP benchmark suites with the dI/dt controller over the baseline machines without any dI/dt control. The Wire-w/Pre configu-

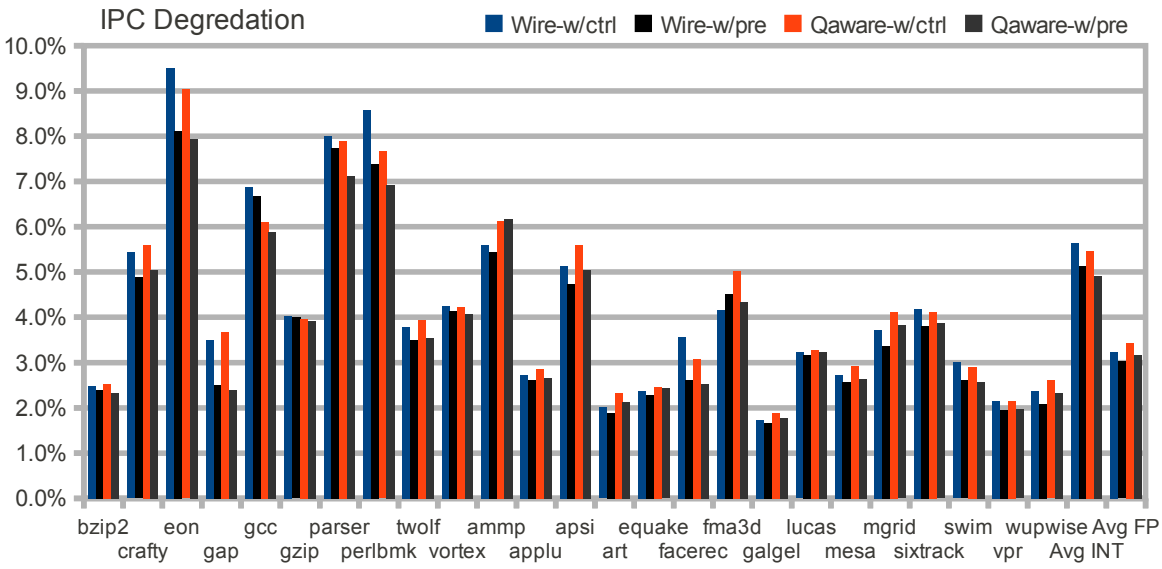


Figure 31. The performance degradation of the dynamic dI/dt controller across the benchmark suite for both the wire-length aware (wire-) and queue-aware (qaware-) floorplans.

ration shows the queue controller with preemptive ALU gating turned on to differentiate the type of applications that can benefit from pre-decoding ALU instructions. Progressive gating in the L2 cache was applied to all cases.

In general, we observe minimal performance degradation for most of the benchmarks. Note that the performance overhead is dependent on the floorplan because it affects the queue configuration. A more optimized floorplan will result in a better-balanced queue configuration. However, if the floorplan results in a configuration where one queue carries a significantly larger number of modules than the others, IPC will be adversely affected due to the fact that the worst-case module activation time is longer. We observe an average performance overhead around 4% for the floorplans that were simulated.

The worst performance degradation is shown by 252.eon, at 9.2% for the controllers without preemptive ALU gating. One explanation for the increased performance impact is due to the fact that the ray-tracing algorithm in 252.eon is ALU intensive. Since modules in the floorplan are asymmetric, the floorplan results in locally clustered ALUs that are not symmetrically distributed in all queues. Highly ALU intensive applications will suffer a performance loss in such cases, since a quick ramp up of modules will take longer if most of the ALUs are clustered in the same queues. A strong indicator of this fact is evident from the higher sensitivity 252.eon shows to preemptive ALU gating, compared with the other benchmarks. Most of the other benchmarks only exhibit little performance loss that is less than 5.7%, overall.

We also observe that preemptive gating of ALUs improves the performance for certain benchmark programs such as 252.eon, 254.gap, 253.perl and 168.wupwise. This is due in part to the fact that the 4-bit decay counter saturates consistently for ALUs (resulting in powering off the module) right before ALU instructions are issued. It is in these scenarios that preemptive gating provides simultaneous performance and dI/dt benefits. The decay counters predict future likelihood of module access solely based on the past activity profile. In contrast, preemptive gating can "look-ahead" and override unnecessary gating that the decay counters themselves cannot prevent, thereby reducing unnecessary performance loss. The minimal IPC overhead illustrates the practical potential of employing a low-overhead technique to control high-frequency dI/dt.

3.5.3.3 Power Consumption Impact

The dynamic controller presented in Section 3.2 includes mechanisms that decrease the frequency of clock-gating events. Clock gating is a technique designed to limit dynamic power consumption. When the dynamic controller prevents a module from turning off to limit dI/dt noise the processor will dissipate some extra energy. When the dynamic controller prevents a module from turning on the processor will dissipate less power, however

the performance impact will cause the program being executed to consume more total energy over an extended period of time. Figure 32 shows the impact on power dissipation for both the wirelength- and queue-aware floorplans using the dynamic controller compared to

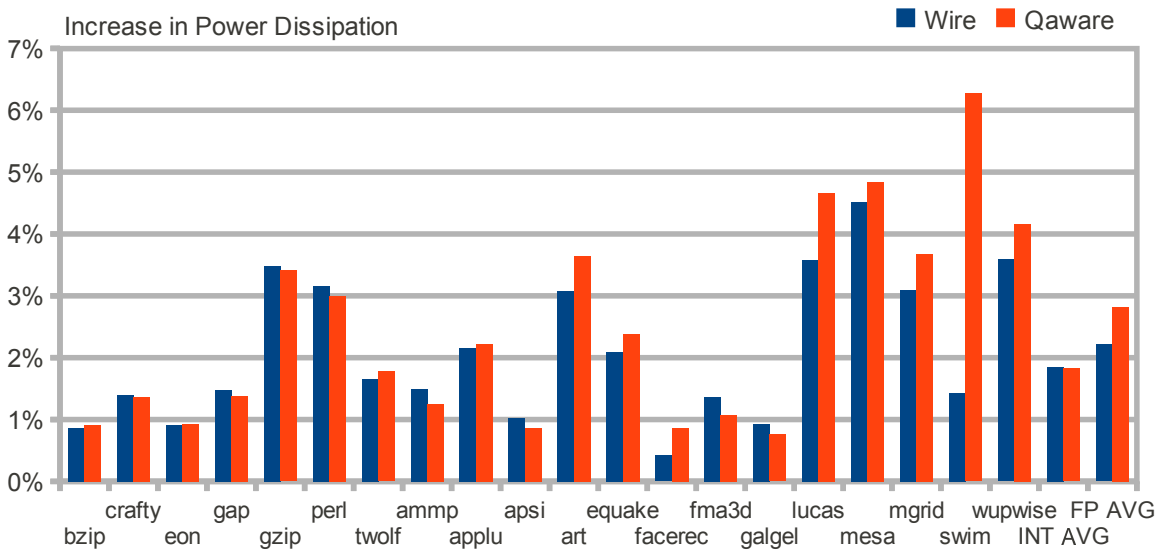


Figure 32. The power consumption impact of the dynamic dI/dt controller on the wirelength- and queue-aware floorplans.

the same floorplans without the dynamic controller. The average increase in power consumption is near 2% for both the floating point and integer suites. The maximum increase in power consumption is 6.3% for the queue-aware floorplan.

3.5.3.4 Thermal Impact

The goal of our technique is to improve power-supply reliability and reduce design effort. Therefore, it is critical that our dI/dt controller must not cause other reliability problems. Since our technique provides fine-grained dI/dt control at the expense of increased power consumption, it is necessary to quantify any potential adverse thermal effects caused by the dynamic controller. Thermal issues are particularly critical in newer processors because of their higher power density, as well as the greater difficulty in dissipating heat across multiple die layers.

We used Hotspot 3.0 [1] to evaluate the thermal impact of our high-frequency dI/dt controller on the given floorplan. We compare our architecture against the baseline design that

uses ideal-clock gating, which represents the scenario with the least power consumption. Figure 33 presents the thermal analysis for all 23 modules in our processor model evaluated

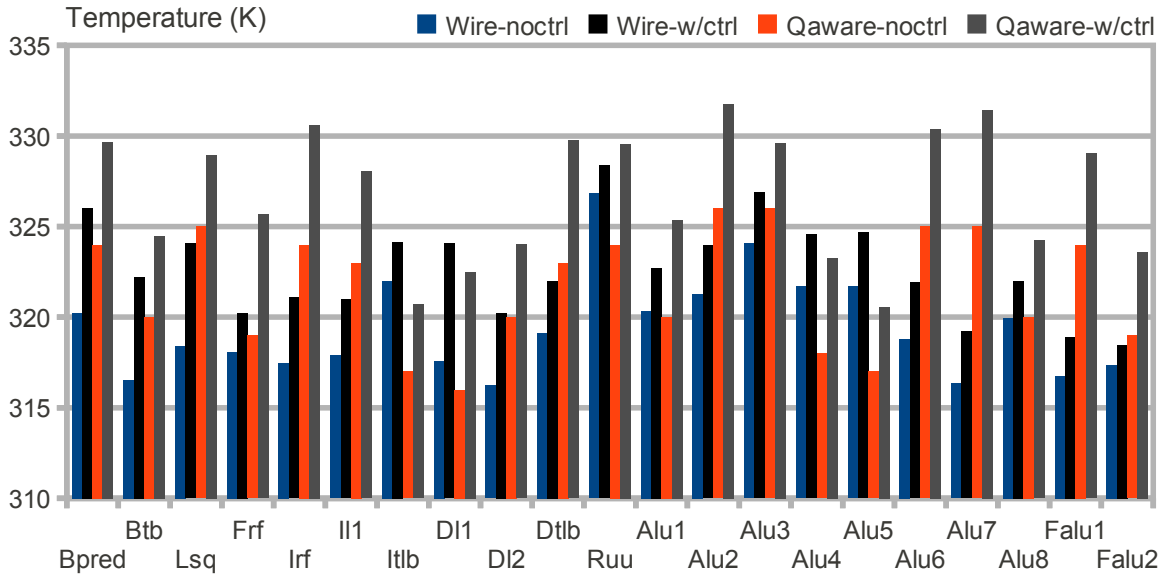


Figure 33. The per-module thermal impact of the dynamic dI/dt controller on the wirelength- and queue-aware floorplans.

using the SPEC2000 benchmark suite for both the wirelength- and queue-aware floorplans.

Overall, we observe nominal thermal impact across all modules. The average temperature increase using the dynamic controller is 3.15 Kelvins for the wirelength-aware floorplan and 5.15 Kelvins for the queue-aware floorplan. Our thermal analysis results indicate that the integration of the dI/dt controller does not create any large thermal problems for our design.

3.5.4 Noise-Direct Floorplan Results

Noise-Direct uses both microarchitectural metrics and module-level current demand to guide optimization. To demonstrate the noise-tolerance of the force-directed floorplan, we compare the Noise-Direct floorplan to a baseline floorplan that minimizes total wirelength. In our noise analysis, we assumed a V_{dd} of 1 volt (for 70nm), and a maximum allowed noise margin of 10%. To illustrate the noise analysis in more detail, we depict the worse-case noise for each module using 164.gzip, a compute-bound program. The noise values are

shown in Figure 34. Note that this graph is sorted from left to right in decreasing order

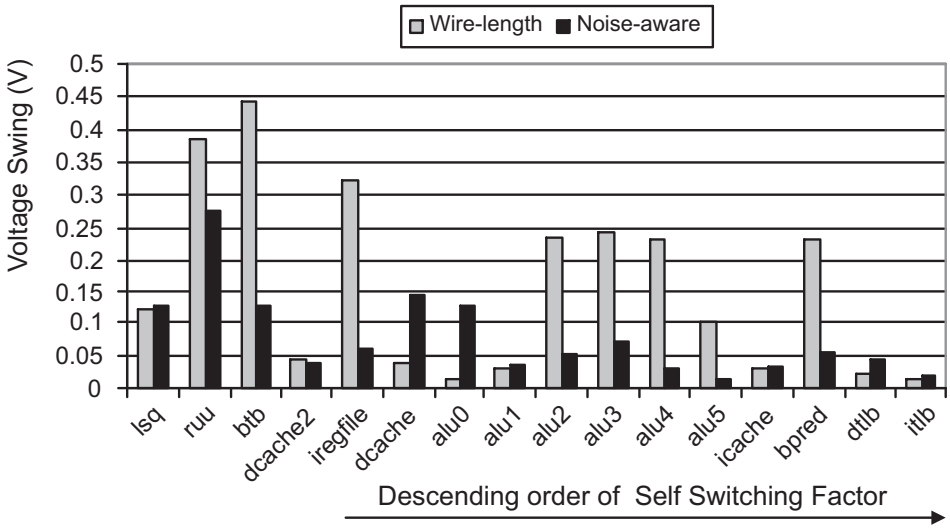


Figure 34. Module-level power-supply noise for 164.gzip **with both the wire-length-driven floorplan and the Noise-Direct floorplan.**

of module self-switching activity. The figure shows that the Noise-Direct floorplan significantly suppresses the noise experienced by modules with high switching activity and high current consumption. Almost all of the ALUs exhibit a fair amount of switching activity, extremely high correlation with each other, and show significant voltage-noise reductions.

For the integer register file (iregfile), the voltage noise was reduced by 81.7% in the Noise-Direct floorplan. We observe that the L1 Data Cache (dcache) and ALU0 have a higher noise violation in the Noise-Direct floorplan as compared to the baseline, although it has a high self-switching factor. This is because other units will have a higher priority for being directed towards power pins because of their strong correlation. The L1 D-Cache does not exhibit a high correlation with other units and hence is less important than other modules that have a higher potential for noise-margin violations. However, it should be noted that the increased violations in the Noise-Direct floorplan are only slightly above the allowed 10% margin, making the overall solution much more noise tolerant.

3.5.5 Queue-Aware Floorplan Results

A comparison of the voltage swing of the controller-aware floorplan and the numbers for Noise-Direct are shown in Figure 35. Queue-aware floorplanning results in overall smaller voltage swing as compared to Noise-Direct. However, most significantly, it reduces the

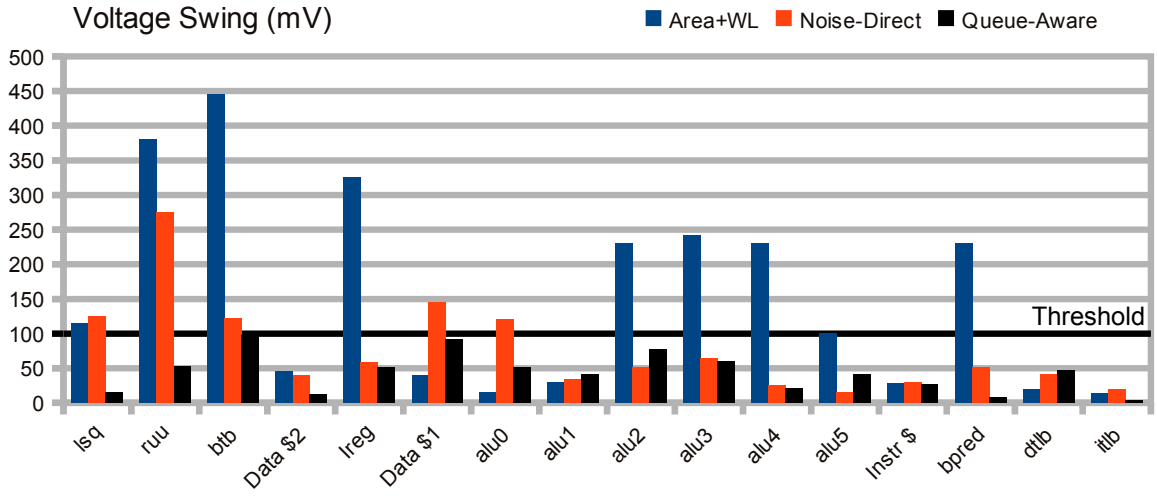


Figure 35. Comparison with Noise-Direct. The voltage violation threshold is 0.1V. This comparison does not include the use of decaps.

voltage swing to be below the 10% voltage violation threshold of 0.1*V* and therefore there are zero noise-constraint violations compared to the approximately 10% noise violations per cycle reported by Noise-Direct. For the purposes of comparison there were no decaps included in the SPICE netlist for these voltage-swing numbers.

Next, a comparison between the traditional area-and-wirelength objective (A+W), floorplanning with *positive Q factor* (+Q), floorplanning *without the queue weights* (No Q), and the new controller-aware floorplanning with *negative Q factor* (-Q), all with decoupling capacitors added, is shown for voltage swing in Figure 36 and noise violations in Figure 37. A comparison between the +Q and -Q bars indicate a change in the cost function switching the Queue factor from positive to negative and shows that our initial intuition about the form of the Queue factor was incorrect. As a reminder, the Queue Factor provides a bonus (in negative form) to the cost function whenever blocks with high correlation and current demand reside within the same dynamic controller queue. The No Q bars show a floorplan

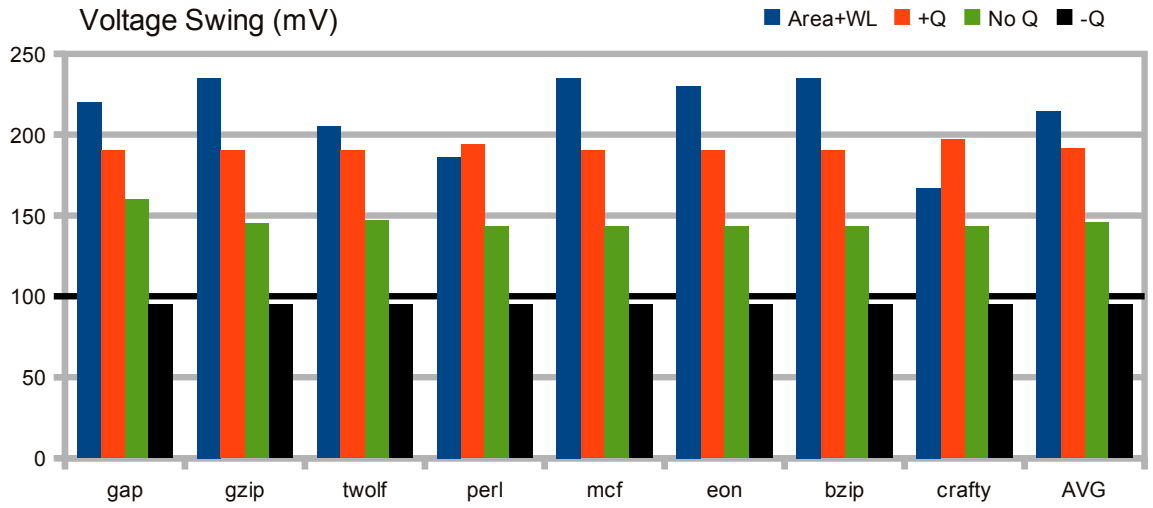


Figure 36. Voltage swing comparison between Area and Wirelength, Positive Queue Factor (+Q), Noise-only (No Q), and Negative Queue Factor (-Q). Decoupling capacitors and the decap allocation network flow are used for these results.

that has 0 for the ϵ weight and thus is most similar to Noise-Direct. As shown in Figure 36, the negative Q controller-aware floorplan has better noise characteristics than those of the traditional A+W objective, the positive Q objective, and the noise-only objective. The queue-aware floorplan has approximately 30% smaller voltage swing than the noise-only objective. This demonstrates that adding queue awareness to the floorplanner has a substantial impact for the simplicity of the change. The negative Q factor floorplan also reduces the voltage swing to be below the violation threshold and therefore there are no voltage violations for this floorplan, as shown in Figure 37. Additionally, the voltage swing graph reveals that the swing is independent of the benchmark for several experiments. This is the result of the dynamic controller fully controlling the coupled voltage swing of the processor. In those cases individual module swings are fully responsible for the magnitude of the chip-level voltage swing. This indicates that the negative Q aware floorplanner is the most effective method to use with the dynamic controller.

Finally, we show that the use of the network-flow-based decap allocation algorithm improves the dynamic noise results. A comparison of the voltage swing between the queue-aware (Queue-Aware) floorplan and the top floorplan according to the cost function (NoFlow) is shown in Figure 38. In the NoFlow case decaps are added in all the white

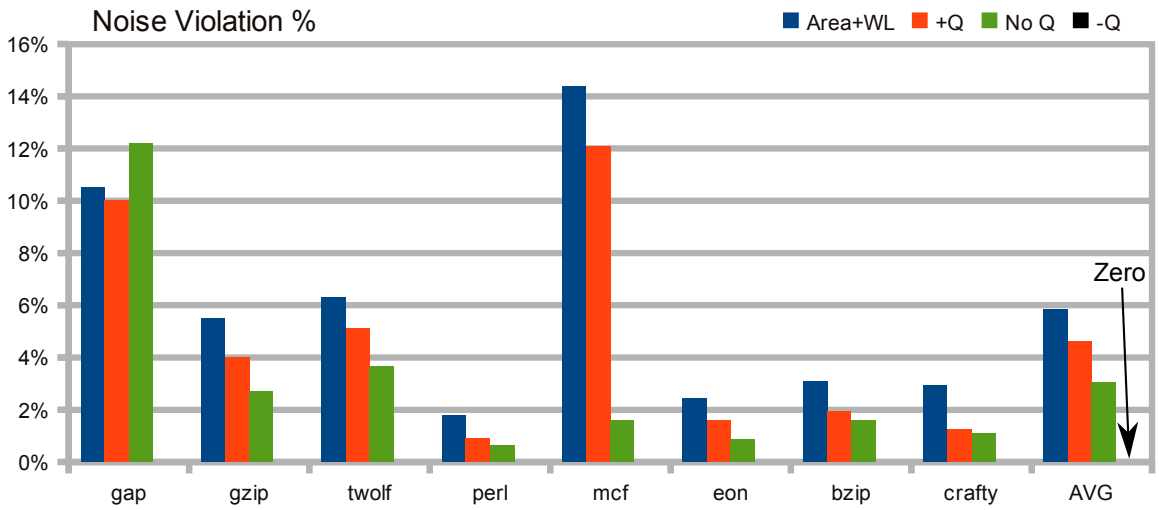


Figure 37. Noise violation comparison between Area and Wirelength, Positive Queue Factor (+Q), Noise-only (No Q), and Negative Queue Factor (-Q). (-Q) has zero violations. As in Figure 36 this data is generated with the decap allocation flow.

space of the floorplan with the lowest cost function value. One can observe that for every

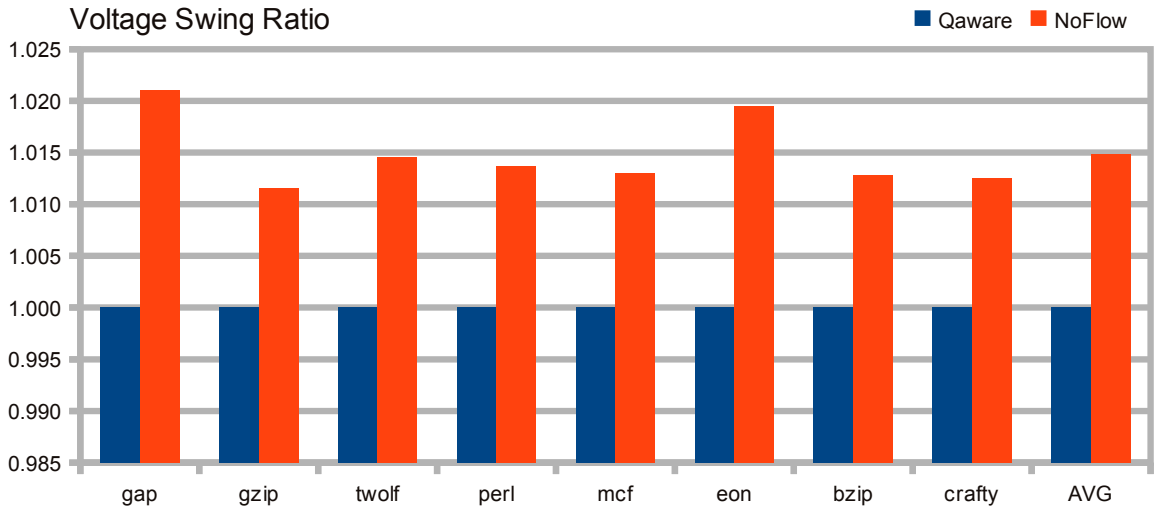


Figure 38. Voltage ratio comparison between using the decap allocation flow (Queue-Aware) and the best according to the cost function (NoFlow). In the NoFlow case the decap allocation flow is not used to choose the best floorplan.

benchmark the floorplan that utilizes the decap allocation flow has improved voltage swing. And in fact, without the use of the decap allocation flow the floorplan does violate the noise threshold by a small amount.

3.6 Summary

The exponential increase in the current consumption of newer generations of processors coupled with aggressive power saving techniques have exacerbated the high-frequency dI/dt issue. If current trends continue, *ad hoc* solutions that mitigate dI/dt effects using excessive decoupling capacitance will eventually become insufficient. Decaps not only occupy considerable chip area but also contribute to the already problematic leakage-power levels. Current microarchitecture-based solutions are inadequate for deep sub-micron designs where high-frequency dI/dt is intricately entwined with both the chip floorplan and power-pin distribution.

This chapter presents a unified design methodology that addresses the high-frequency dI/dt issues and maintains high reliability while alleviating the design cost of creating a low impedance power delivery network using a dynamic queue-based dI/dt controller and a controller-aware floorplanning algorithm. By leveraging microarchitectural profile information in the floorplanning stage and monitoring application-based module activity at run-time with our dynamic controller, we show that current demands can be guaranteed for modules residing within the same power-pin domain. In addition, we integrate a preemptive ALU gating mechanism as a performance enhancement technique as well as an enhanced progressive gating technique for large modules (L2 cache) into our queue-based control mechanism. We have also explained how the dI/dt architecture can be implemented in a conventional out-of-order pipeline in a complexity-effective manner. Experimental results show that our dI/dt controller can improve the current variability of applications by an average of 7x with a mere 4.0% IPC degradation and very little power consumption increase. SPICE simulations show that the combination of our dynamic controller and controller-aware floorplanner can completely eliminate power-supply noise-margin violations.

Overall, our design provides a practical microarchitectural technique and a physical design approach that can be used in concert to alleviate the effort of design afterthoughts and reduce the use of leaky decoupling capacitors that consume larger chip area. Our

technique incurs little performance overhead and has very little thermal impact.

CHAPTER 4

MANY-TIER 3D POWER-SUPPLY-NETWORK SCALING

The shift to the multi-core era has increased the memory-bandwidth demand of high-performance processors. 3D integration of memory and processors has emerged as a potential solution to this memory bandwidth problem. However, there are several unanswered questions regarding 3D integration. This chapter presents a study of integration scaling from a power-supply and thermal perspective, as well as two design techniques to lower supply noise in large-scale 3D systems. As the number of tiers in a 3D system increases, the uppermost tiers become separated from the power-supply bumps by increasing resistance and inductance. The cause of these parasitics are the through-silicon vias (TSVs) that provide communication between tiers. Additionally, increasing integration quickly raises the volumetric power density of the IC stack.

Power-supply noise in ICs is caused by several factors. The largest factor creating transient noise is the so-called "first droop" noise that results from the interaction of the inductance in the package and the on-chip decap during sleep transitions. Power gating due to sleep transitions causes large transient changes in the current demand on the power-supply network. In the case of 3D systems, the TSVs add inductance of their own to the inductance that naturally exists in the package.

Thermal impacts are another major factor that must be considered when designing 3D systems. Standard air-cooled heatsinks will be unable to cope with the high volumetric power density of large-scale 3D systems. Recent work has focused on implementing micro-fluidic channels [101] onto the backs of 3D-stacked ICs for the purpose of removing heat using liquid-phase fluids. The heat removal capacity of these systems is very promising. In this work we assume that micro-fluidic heatsinks are used to dissipate heat, and design our power distribution model around this assumption.

Previous work on power-supply issues related to 3D stacking has examined the problem mainly from a packaging perspective [69, 97]. Other works that have investigated the impact of 3D stacking have limited their scope to systems with only a small number of tiers [24, 102, 103]. In this chapter we present detailed transient simulation of 3D stacks with up to 46 tiers and examine issues related to scaling various components of the power distribution grid topology, as well as scaling of other important components related to power integrity in large-scale 3D systems. Additionally, we provide thermal results using micro-fluidic heatsinks for the same systems.

4.1 3D and Flip-Chip Power Nets

High performance 3D systems will generally use flip-chip-style packaging to increase off-chip interconnect density and reduce parasitics. Flip-chip power-distribution systems are commonly laid out as grids [39]. High-level metal layers are reserved for laying out a coarse-grained grid with large wires that connect a regular array of power and ground C4 bumps. A fine-grained mesh provides local distribution and connects to lower-level-metal power rings or standard-cell-row distribution wiring. Most commercial products today have C4 bump pitches around 100 to 200μm [39], however, researchers have demonstrated micro-bumps with pitches below 50μm [104].

The TSVs will fill the role of the C4 bumps for intermediate tiers in 3D systems. Each tier will contain its own power-distribution network. Figure 39 shows the general topology of a 3D power distribution network without the C4 bumps. The vertical resistance between adjacent tiers should be close to that of the C4 bumps to maintain reliable power and ground voltages in large-scale 3D systems. The resistance of individual C4 bumps is on the order of 5$m\Omega$. Additionally, TSVs should be smaller than the C4 bumps or large amounts (25% or more) of die area will become unusable. Accordingly, power-delivery TSVs should be continuous or should be connected serially with large-numbers of local vias if continuous TSVs are not available.

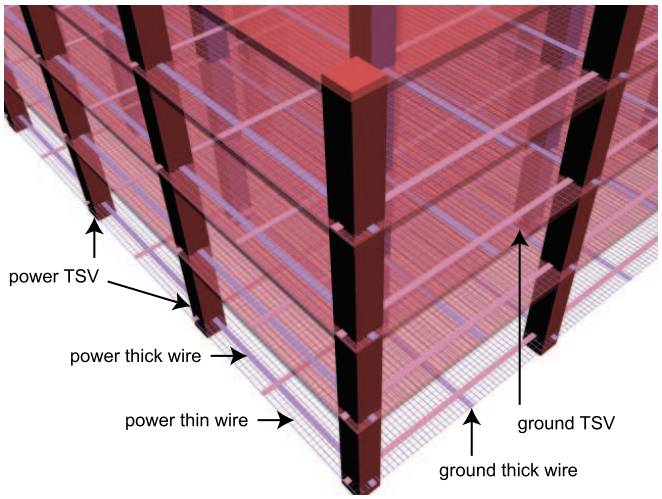


Figure 39. Wires and TSVs in a 3D P/G network

TSVs can be manufactured in many different sizes. Diameters near $1\mu m$ have been shown in the literature [105]. Power and ground TSVs should be large to have low resistance, but signal TSVs should be small to increase interconnect density and reduce parasitic capacitance. Manufacturing multiple TSV sizes on a single die would require multiple processing steps and thus increase cost and reduce yield. Therefore, it may be necessary to use a single TSV size for both power distribution and signal wiring.

There are several potential combinations of TSV size and distribution that could be used to deliver power. Figure 40 shows some of the basic choices. The figure depicts three TSV topologies for a single cell in the power/ground network. This cell is tiled all over the chip.

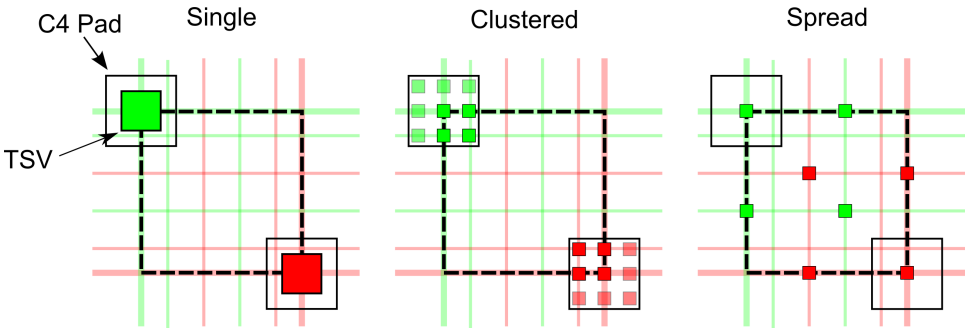


Figure 40. Three TSV topologies for power (red) and ground (green) distribution in a single cell of the distribution network. The combined resistance of all TSVs in each topology is equal.

In the *single* topology one large TSV is centered over the C4 pads for both power (red) and

ground (green) distribution. The *clustered* topology clusters multiple small TSVs around the C4 pad. Finally, the *distributed* topology spreads small TSVs evenly throughout the die. For each of the topologies the combined resistance of all the TSVs is the same because the total cross-sectional area of the TSVs is the same.

4.1.1 TSV Topology Comparison

There are four main metrics used to compare the TSV topology options. The metrics are area overhead, inductance, capacitance, and power-supply-noise performance. Our baseline power grid dimensions use a power-to-power C4 pitch of $400\mu m$. The ground bumps are offset from the power bumps by $200\mu m$ in both the x and y direction. The baseline single TSV size is $40 \times 40\mu m$. The area overhead of the single topology with this size TSV is therefore 2%. The clustered topology uses $13 \times 13\mu m$ TSVs with an area overhead of just over 7%. Finally, the distributed topology has an area overhead of around 3.3%.

We simulated the inductance of the three topologies using Synopsys' inductance extractor, Raphael [106]. We consider the mutual inductance between the power and ground TSVs along with their self inductance to measure the effective inductance of the individual TSVs as well as the effective inductance for each tile. Table 7 shows the effective inductance values obtained from our simulations. In general, smaller TSVs have higher inductance, however, the clustered and distributed topologies include multiple TSVs in parallel, which results in a lower effective inductance. The large distance between the TSVs in

Table 7. Effective inductance values in pH for power distribution TSVs. The TSV lengths are in μm.

	TSV length	
Cross Section	$15\mu m$	$150\mu m$
Clustered $13 \times 13\mu m$	0.496	20.42
Distributed $13 \times 13\mu m$	0.308	6.63
$20 \times 20\mu m$	2.234	69.65
$40 \times 40\mu m$	1.276	50.80
$60 \times 60\mu m$	0.882	40.49
$80 \times 80\mu m$	0.666	33.62

the single topology cause it to have the highest effective inductance. The clustered topology has a larger footprint that results in smaller loop area between the power and ground TSV arrays. Additionally, the outer members of the cluster provide inductive shielding to the inner members, again lowering inductance. In the distributed topology there are much closer return paths between power and ground TSVs, which results in the lowest inductance of the three topologies.

We again use Raphael to simulate the capacitance of the three topologies. Here we are most concerned with the coupling capacitance between the power/ground TSVs and nearby signal TSVs. In this case the clustered topology creates the largest coupling capacitance. This is a result of the increased sidewall area exposed to the signal TSVs in the worst case. The distributed and single topologies have similar capacitance, with the single topology having slightly higher values.

Finally, we examine the power-supply-noise performance of the three topologies using a simple uniform power distribution model. The dynamic and DC per-tier uniform power distributions on a single tile of the power grid is simulated. Figure 41 shows a comparison of the noise values for the three topologies. Our simulations show that there are two

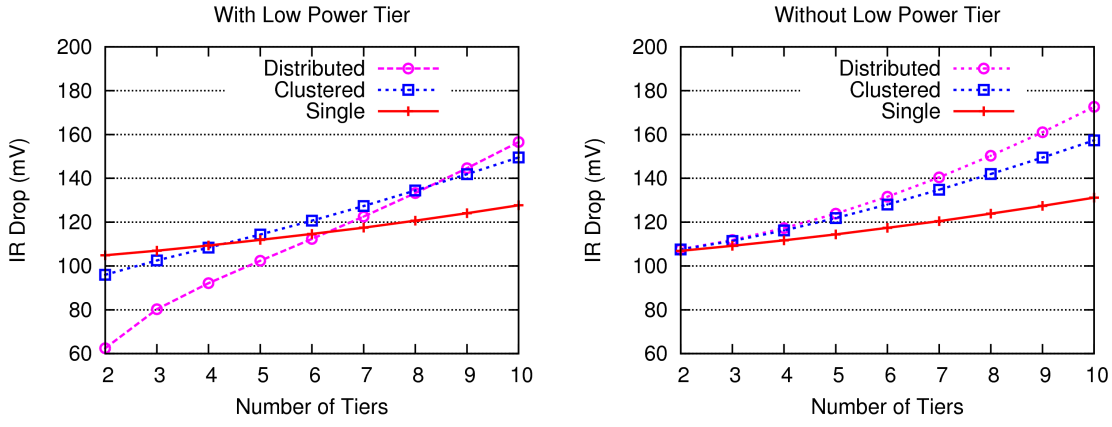


Figure 41. A comparison of the IR drop scaling for a simple 3D power distribution grid among the three TSV topologies. The left graph shows the case where the top tier has much lower power dissipation than the other tiers. In the right graph all the tiers have equal power dissipation.

main considerations. For dynamic noise, which has the largest magnitude, the reduced

inductance of the clustered and distributed topologies result in the lowest noise values. Secondarily, for systems with low power tiers it is possible for the distributed topology to deliver power through their distribution networks to the high power tiers, which results in lower noise values. However, for uniform or nearly-uniform stacks the clustered topology always results in lower noise than the distributed topology. Results for full-chip simulations of the three TSV topologies are presented in Section 4.5.

4.2 TSV RLC Parasitic Modeling

The focus in this work is on TSV parasitics as they apply to the power distribution network. In the power distribution network it is more important to reduce parasitic inductance and resistance than to save space. For our base case, and the limit study which is to follow, we assume copper TSVs with a square cross-section of $40 \times 40\mu m$. For scaling, we consider TSVs that range in size from $20 \times 20\mu m$ to $80 \times 80\mu m$. We also consider several TSV distributions using smaller-size TSVs. The length of the conducting path for our TSVs, equivalent to the thickness of the die through which they pass, is assumed to be $15\mu m$ for thinned dies and $150\mu m$ for dies that contain micro-fluidic channels or are not thinned substantially.

4.2.1 Resistance Scaling

The resistance of a metal interconnect is calculated assuming a uniform current density. This assumption is valid in the power-supply grid because there is no high-frequency oscillation that would cause skin-effects to become dominant, as could be the case in signal TSVs. The resistance of a TSV, R_{TSV}, is calculated as:

$$R_{TSV} = \frac{l \cdot \rho}{A}. \tag{11}$$

In this equation, l is the conducting path length, ρ is the resistivity of the TSV material, and A is the cross-sectional area of the conducting path.

In this work we assume that TSVs are made from copper and we use a conservative

estimation of the resistivity of $21n\Omega \cdot m$. This value should account for any thermal effects. Additionally, the sizes of the TSVs we are investigating are large enough that the resistivity should approximate the behavior of bulk copper. Table 8 shows values for the resistance of some typical TSVs in the power distribution networks simulated in this chapter. The table

Table 8. Resistance and capacitance values in $m\Omega$ and fF, respectively, for typical power distribution TSVs. All lengths are in μm.

	Resistance TSV length		Capacitance TSV length	
Cross Section	$15\mu m$	$150\mu m$	$15\mu m$	$150\mu m$
$20 \times 20\mu m$	0.788 $\mu\Omega$	7.875 $\mu\Omega$	14.28 fF	142.80 fF
$40 \times 40\mu m$	0.197 $\mu\Omega$	1.969 $\mu\Omega$	26.60 fF	266.02 fF
$60 \times 60\mu m$	0.088 $\mu\Omega$	8.750 $\mu\Omega$	38.93 fF	389.33 fF
$80 \times 80\mu m$	0.049 $\mu\Omega$	0.492 $\mu\Omega$	51.25 fF	512.53 fF

shows that the resistance values for the thinned die can be very low, less than one milliohm.

4.2.2 Inductance Scaling

The main cause of low-frequency first-droop power-supply noise is the interaction between the inductance of the package and the capacitance on-die. By adding large, vertical TSVs for power delivery there is the chance that the power-supply noise problem may be exacerbated. The TSVs in the power-supply network have a much larger pitch than length, so the mutual-inductance of neighboring TSVs is dominated by the self-inductance of the TSVs.

The self-inductance of a TSV, L_{TSV}, with rectangular cross-section can be calculated approximately using the following equation [107]:

$$L_{TSV} = 2 * l \left[ln \frac{2 * l}{w + t} + 0.5 - k \right] \tag{12}$$

where $k = f(w, t)$ and $0 < k < 0.0025$, l, w, and t are the TSV length, width, and height in cm, respectively, and L_{TSV} is measured in nH. However, this formula breaks down when the length of the conducting path is smaller than the other two dimensions.

Figure 42 shows a plot of the total inductance seen by the top tier for different numbers of stackings of our scalable prototype system (discussed in Section 4.4). Each colored line represents a TSV of a different dimension. The inductance values are computed by

Synopsys' inductance extractor, Raphael [106], and include the contributions from both self and mutual inductance. The solid black bar across the graph represents the inductance of the package apportioned to one bump in our simulations. The graph shows that TSVs in large stacks can more than double the inductance value seen by tiers that are furthest from the bumps. Table 7 shows the effective inductance values used in our power-noise simulations. The individual effective TSV inductance for the clustered and distributed TSV topologies can be obtained by multiplying the values in the table by nine.

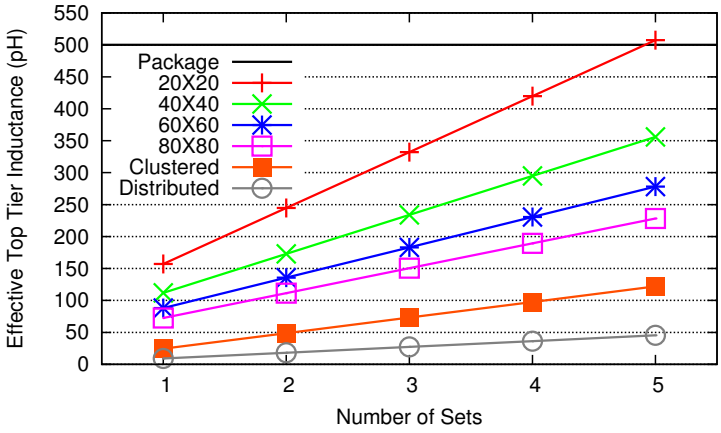


Figure 42. Inductance scaling for various TSV dimensions. The solid bar represents the inductance in the package apportioned to one bump. Large set stackings can nearly match the inductance of the package bumps for the tiers farthest from the bumps.

4.2.3 Capacitance Scaling

TSV capacitance can improve the dynamic-noise performance of the power-supply network. However, we provide TSV capacitance values here for the sake of completeness. The capacitance of the power/ground TSVs comes solely from horizontally neighboring wires and TSVs. Capacitance can be calculated by breaking the problem into several parts.

The capacitance of two TSVs placed in parallel is calculated as follows:

$$C_{TSV,p} = \epsilon_{di} \frac{(H_{TSV} - 2H_{INT})W_{TSV}}{S_{TSV}}, \tag{13}$$

where ϵ_{di} is the dielectric constant of silicon dioxide, H_{TSV} is the TSV height, H_{INT} is the height of the TSV covered by surrounding signal wires, W_{TSV} is the cross-sectional dimension of the TSV, and S_{TSV} is the separation between the TSVs. The coupling capacitance

of two diagonally placed TSVs is calculated as follows:

$$C_{TSV,d} = \epsilon_{di} \frac{1}{\pi \sqrt{2}} \cdot H_{TSV} K_{corner}, \tag{14}$$

where K_{corner} is an empirically-derived constant dependent on geometry and spacing. The capacitance of the TSV due to surrounding wires is calculated by:

$$C_{TSV,sw} = \frac{m_{sw} - 1}{2}(C_{side,1} + C_{side,2}) + 4(C_{side,3} + C_{side,4}), \tag{15}$$

where m_{sw} is the number of metal layers, and $C_{side,i}$ are the sidewall capacitances of the various sidewalls of the TSVs. Finally, total TSV capacitance is calculated as follows:

$$C_{TSV} = 4(C_{TSV,p} + C_{TSV,d}) + C_{TSV,sw}. \tag{16}$$

Detailed calculation of sidewall TSV capacitance is complicated and requires consideration of neighboring wires in many orientations. The values used in our experiments for capacitance were computed using the formulas above, as well as with Raphael [106], and are shown in Table 8.

4.3 Many-Tier Prototype System

The system targeted by this work consists of a multi-core processor with system memory (DRAM) integrated onto the same 3D stack. This also implies the integration of a memory controller. We chose to use the Intel Penryn [108] architecture as a baseline. In the past, scaling of system memory followed processor speed. In the multi-core era system memory scaling will likely follow number of processors instead. Taking this into account, we assume a constant ratio of 2 GB of DRAM per processor core. Using the next-gen memory-density assumptions of Loh [103] we create a "set" of our scalable prototype system that consists of a single tier of processors and eight tiers of DRAM. Each tier of processors contains four cores and each tier of DRAM contains 1 GB of storage. This "set" can be stacked any number of times along with memory controllers to create a scalable many-tier system. This system is assumed to have a footprint of approximately 300 mm^2.

Our initial target was to examine a system composed of a large number of tiers. Most previous works consider only two to four tiers, with a few works considering up to eight. We chose to examine a system with approximately 50 tiers. Using our scalable prototype design we choose five "sets" that, along with a memory controller tier, provide a system containing 46 tiers. Using the power consumption assumptions described in the next section, this 46-tier system would consume approximately 525 watts. Unlike systems designed from the ground up to consider 3D integration technology, this prototype has a relatively low signal TSV requirement. Only signals that are currently sent off-chip need to be sent between tiers. Given the much finer possible pitch of TSVs with respect to off-chip bumps, this does not stress the limits of TSV fabrication technology.

The system that is targeted in this work contains many components that dissipate a large amount of power. Combined with the stacked nature of 3D technology, the heat removal path through a standard air-cooled heatsink would be far too thermally-resistive to support full-speed operation of the system. Accordingly, we assume the use of efficient micro-fluidic heatsinks [101]. We assume that each set of our scalable prototype contains a group of micro-fluidic channels fabricated onto the back of the processor tier. Typical 3D-stacked circuits have dies that are thinned to very small thicknesses, in the range of 5 to 20 μm. However, to accommodate the size and mechanical stresses of micro-fluidic channels, the tiers that contain them are assumed to have a thickness of 150 μm, while others have a thickness of 15 μm.

An important consideration when designing a large-scale 3D system is the organization of the different types of tiers. Various tiers have differing power consumption, noise-generation, and decoupling capacitor distribution profiles. All of these factors affect the thermal and noise profiles in various ways. In this work, we consider three simple organization styles. These organization styles are detailed in Figure 43. In the cores-first organization style the processor tiers are all placed next to the bumps. For the cores-spread organization style the processor tiers are separated by the memory tiers associated with

each set. Finally, the cores-last organization style places the processor tiers closest to the air-cooled heatsink, but farthest from the bumps.

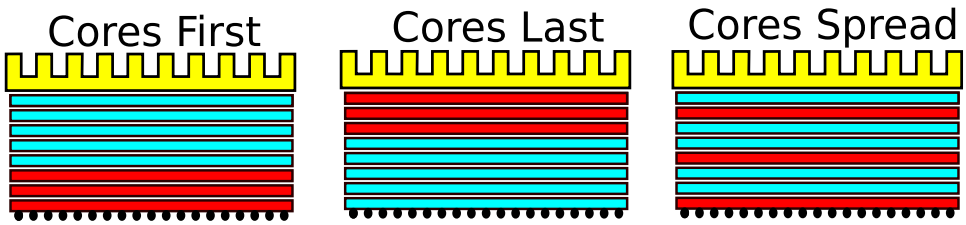


Figure 43. Core organization styles. The red bars represent processor tiers and the blue bars represent memory tiers. Each style has differing power-noise and thermal characteristics.

4.4 Modeling

Our scaling studies are targeted at a combination multi-core processor and DRAM system. There are several major sections of the final power-supply-noise model. This model includes the power-supply grid and the power-consumption maps for the processor, memory, and memory-controller tiers. We also include decoupling capacitors (decaps) in a uniform distribution. The decaps on the memory tiers are assumed to have one quarter the density of those on the processor or memory controller tiers.

4.4.1 Power Maps

The processor power map used in this work is based on the Intel Penryn architecture. We examined a publicly released die photo of the 45 *nm* Penryn and produced a floorplan based on the work from Puttaswamy and Loh [109] to create this map. We then divided the logical pipeline stages of the floorplan by area, and generated the power distribution. This distribution was created by dividing a total power dissipation of 54 watts for a dual-core version into the component stages for two processors. The percentage breakdown for each stage are based on the numbers published by Intel's George *et al.* [108].

The power map for the DRAM tiers is broken into two parts. For worst-case noise behavior we model the IDD07 condition [110], all banks interleaved read current, which is widely acknowledged as the condition generating the most power-supply noise. For static

IR-drop calculations we use an average system power value generated from Micron's system design power calculation spreadsheet. In both cases the values used are projected from currently available DDR3 data to the size of the memory modeled in our targeted processor plus DRAM system. The values used in this work for the DRAM tiers are provided in Table 9.

Table 9. Power map characteristics of DRAM and memory controllers used in our simulations.

	DRAM	Memory Controller
Rise-Time (ns)	20	5
Current Demand (A)	0.6	10
Active Area (mm^2)	27	50
Power Consumption (W)	0.6	10

The power map for the memory controller tier is based on data sheets released for various north-bridge chips. There is very little power consumption information available for just memory controllers. In this case we conservatively estimated that 50% of the average power consumption for a north bridge is from the memory controller. We also estimate that the area of a single memory controller is about 50% of the total chip size. The values used in this work for the memory controller tier are shown in Table 9.

4.4.2 Power Grids

The scaling studies presented in the next section vary many of the design parameters of the power grid. The values presented in this section represent the baseline case. The power grid for the processor tiers contains two levels of granularity. We assume a flip-chip package with ball-grid-array chip connections. The power/ground TSVs are connected directly to the power/ground balls and travel vertically through the stack to the top tier. The TSVs are connected to one another in a grid pattern using thick 10 μm wires. Within this large coarse mesh is a fine-grained mesh for local power delivery. There are 20 of these small 5 μm wires for each of the power and ground grids in each direction. The power-to-power pitch of the bumps, TSVs, and coarse grid is 400 μm. The ground grid is offset from the power grid by 200 μm in each direction. This allows the power TSVs to accommodate the

baseline size and pitch of the micro-fluidic channels. The memory controller tiers have the same power distribution grid as the processor tiers.

The DRAM tiers have a power grid that is again based on two levels of granularity. They share the same coarse grid that the processor tiers contain. However, due to the lower power requirements of the DRAM tiers we assume that their fine-grained power distribution wires are at half the density of the processor tiers, that is, there are 10 small 5 μm wires for each of the power and ground grids. This effectively causes the DRAM tiers to have one quarter the power/ground metallization that the processor tiers have.

4.4.3 Circuit Models

To model the 3D power distribution grid, we assume that TSVs and package bumps have parasitic resistance, capacitance, and inductance. The 2D distribution grid that exists on each tier of our system is purely resistive. A capacitor (representing decap) and a current source (representing the current demand of the transistors) connects the power and ground grids at each node. The current sources are simulated as a ramp from zero to the current demand of the particular module that covers that area of the floorplan. The rise-time of the current-source ramp is dependent on the type of tier (processor, DRAM, or memory controller) that the current source is located on.

4.4.4 Power-Noise Simulation

Simulation of power grids is a current topic of research because of their large size. Typical power grids contain millions of circuit nodes. Most commercial and free circuit simulation tools cannot efficiently simulate circuits of this size. To deal with this problem, we created a custom circuit simulator. This simulator is based on Modified Nodal Analysis [111] (MNA). We also add a modification based on Domain Decomposition [112] (DD).

Domain Decomposition is a technique used mainly for solving partial differential equations. For circuit analysis, the basic idea is to designate different domains of the circuit based on their connectivity and then an interface between all domains. For our simulations

the TSVs provide a natural (and low node-count) interface boundary. Matrix equations from MNA for the individual domains are then solved and combined with the matrix equations for the interface to arrive at the final solution.

$$[A] \rightarrow \begin{bmatrix} A_1 & 0 & E_1 \\ 0 & A_2 & E_2 \\ F_1 & F_2 & A_\gamma \end{bmatrix}$$

Solving $Ax = b$ then devolves to solving

$$\{A_\gamma - F_1 A_1^{-1} E_1 - F_2 A_2^{-1} E_2\} \cdot y = g - F_1 A_1^{-1} b_1 - F_2 A_2^{-1} b_2$$

$$x_1 = A_1^{-1} b_1 - A_1^{-1} E_1 y$$

$$x_2 = A_2^{-1} b_2 - A_2^{-1} E_2 y$$

where

$$x = \begin{pmatrix} x_1 \\ x_2 \\ y \end{pmatrix} \quad b = \begin{pmatrix} b_1 \\ b_2 \\ g \end{pmatrix}$$

The total time for matrix inversion is thus lowered by using the DD technique because matrix inversion is an $O(n^3)$ operation.

Using these techniques (MNA with DD) we have simulated networks containing up to 11 million nodes. Unfortunately, our full-size 46-tier system contains over 25 million nodes. To efficiently simulate our largest-size system we therefore limit our studies to an area that covers one core. This area contains one core per processor tier as well as one memory controller and the active areas of the DRAM tiers. Our experiments indicate that this introduces a small amount of error, approximately 5%. However, the error is systematic in nature and should not affect the conclusions of the scaling studies.

This work focuses on a conservative approximation of the power-noise of a given system. Accordingly, our dynamic noise simulations are of an extremely worst-case scenario. We simulate the power noise generated from *all* processors in the system being powered

on from a sleep state at one time at the same time that every tier of DRAM is continuously in the interleaved-read state.

4.4.5 Thermal Model

The three dimensional thermal model of Koo *et al.* [113] is modified for our work to consider the lateral temperature and flow-rate distribution caused by a non-uniform power/heat flux distribution. It is assumed that the temperatures of both the fluid and solid are uniform within each grid point. The thermal conductivity of the oxide metal layer is conservatively assumed to be the same as that of the oxide layer. In fact, since those layers are very thin, and have low thermal conductivity compared to the silicon layers, heat transfer through these layers is negligibly small, which validates the assumption. Thermal and fluid flow in the micro-fluidic channel, the energy and momentum conservation equations for fluid flow, and the energy conservation equation for solids are established and given by,

$$\dot{m}\frac{di}{dz} = \dot{q}_{conv} = \eta_o h_{conv}\tilde{P}\left(T_{w,k} - T_{f,k}\right) + h_{conv}w\left(T_{w,k+1} - T_{f,k}\right) \tag{17}$$

$$-\frac{dP}{dz} = \frac{2fG^2}{d_h\rho} \tag{18}$$

$$\frac{\partial}{\partial x}\left(k\frac{\partial T_w}{\partial x}\right) + \frac{\partial}{\partial y}\left(k\frac{\partial T_w}{\partial y}\right) + \frac{\partial}{\partial z}\left(k\frac{\partial T_w}{\partial z}\right) + \dot{q}_g + \dot{q}_{conv} = 0 \tag{19}$$

In these equations, T_w and T_f represent the temperatures of the solid and fluid, respectively, $\dot{m}$, i, and h_{conv} are the mass flow rate, enthalpy, and convective heat transfer coefficient of the fluid, respectively. For the micro-fluidic channel, heat is directly transferred only to the channel base. The channel wall is modeled as a fin attached to the base. Thus, the overall surface efficiency (η_o) is adopted to characterize an array of fins and the base surface. Micro-fluidic channel geometry information is given by channel perimeter, $\tilde{P}$, and width, w, and the irregularity of the micro-fluidic channel layer placements are also provided into the set of thermal equations to incorporate the effect of the number of micro-fluidic channel layers.

Equation (17) represents that the fluid enthalpy gradient is brought by convective heat transfer, $\dot{q}_{conv}$, which is a consequence of the temperature difference between the solid and

fluid, as well as fluid convective motion. The pressure drop across the micro-fluidic channel can be obtained from the fluid momentum conservation equation (18). Terms P, G, and ρ are the pressure, mass flux, and density of the fluid; f is the fluid friction factor and d_h is the hydraulic diameter of the micro-fluidic channel. Equation (19) is the three dimensional thermal conduction equation with two source terms, i.e., heat generated from the active and oxide-metal layers, $\dot{q}_g$, and convective heat transfer, $\dot{q}_{conv}$. Finally, k represents the thermal conductivity of the solid, in this case silicon.

Deionized water is the working fluid and is assumed to be in a single-phase inside the micro-fluidic channel. The governing Equations (17), (18), and (19) are integrated and then discretized using the upwind scheme [114]. The discretized equations are simultaneously as well as iteratively solved using the successive under-relaxation (SUR) method to deal with the non-linearity of the system. The properties of water are determined using REFPROP 6.0 [115].

4.5 Experimental Results

Figure 44 shows the voltage drop as a function of time for our five-sets-stacked cores-first

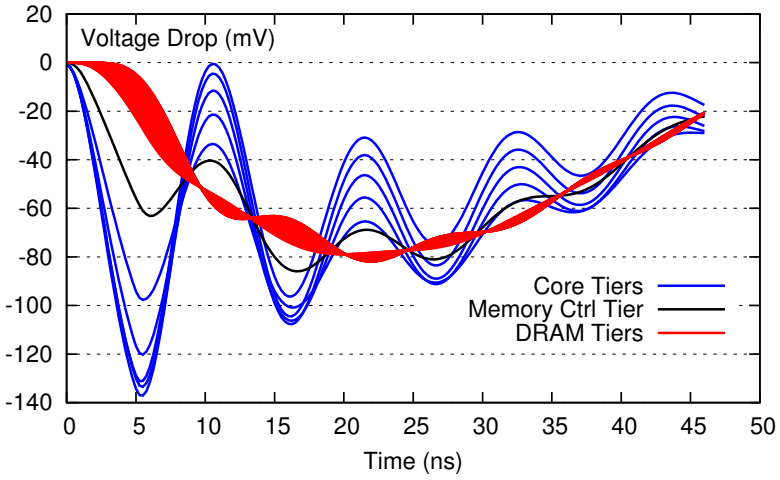


Figure 44. An example of voltage drop as a function of time. Each line represents the noise at one point on one tier.

organization style. Each line in the graph represents the noise at one point for one tier. The figure shows several interesting features. The noise of the core tiers has a much larger

amplitude than that of the memory tiers and the frequency is also higher, corresponding to the 5 *ns* rise-time of the core tiers. However, the voltage drop of the entire system follows the 20 *ns* rise-time of the memory tiers.

4.5.1 Effect of TSV Inductance

Figure 45 shows the % error introduced by ignoring the TSV inductance during dynamic power-noise analysis. The error is generally low when there are a small number of tiers

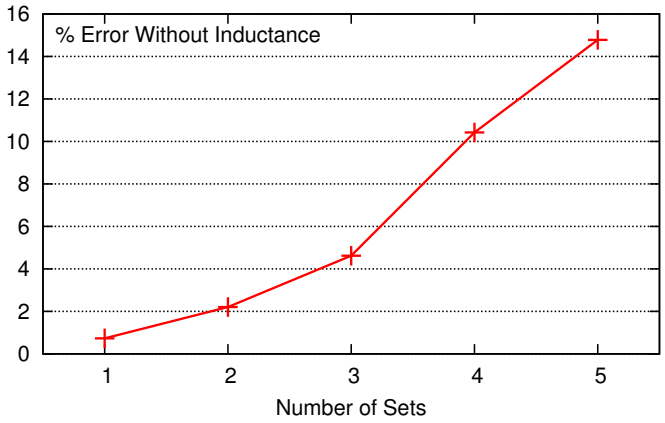


Figure 45. The % error introduced by ignoring TSV inductance during dynamic noise simulations.

stacked together. However, as the tier count of the stack rises the % error goes much higher, with the largest case examined here reaching an error of 14.8%. For the tight noise constraints of modern power-supply designs this error may cause unexpected failures resulting in multiple respins.

4.5.2 Power-Noise Scaling Results

For clarity of presentation, all results in this section relate to the cores-first organization style. Results relating to the other organization styles are presented in Subsection 4.5.3. Figures 46 and 47 show the effect on dynamic noise and IR drop of scaling the TSV dimension from $20 \times 20\mu m$ to $80 \times 80\mu m$. Larger TSVs provide lower resistance and inductance values, as shown in Section 4.2. The TSV dimension has a much stronger effect on the IR drop than dynamic noise. However, the effect becomes more and more pronounced as the number of sets increases.

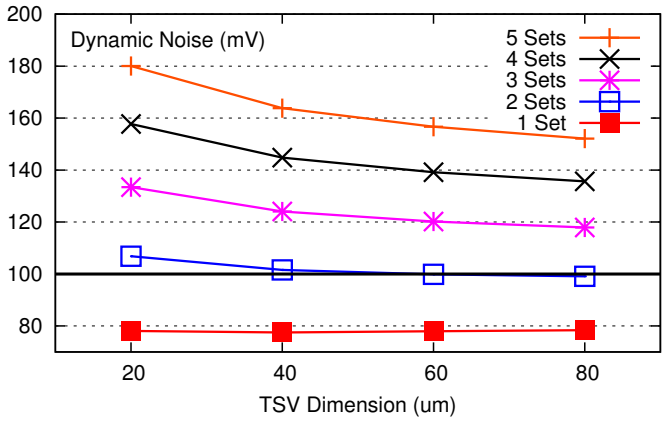


Figure 46. The dynamic noise as a function of TSV dimension in μm. The baseline TSV dimension is $40 \times 40 \mu m$.

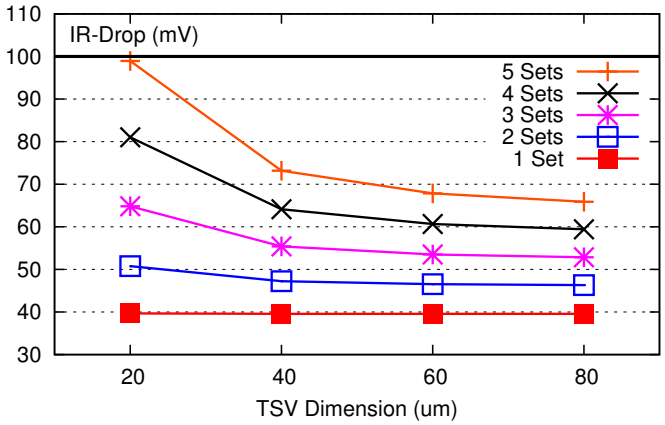


Figure 47. The IR drop as a function of TSV dimension in μm. The baseline TSV dimension is $40 \times 40 \mu m$.

In Figures 48 and 49 we show the effects of simultaneously changing the TSV and bump pitch on dynamic noise and IR drop. Both the dynamic noise and IR drop are highly influenced by the pitch of the TSVs and bumps. Comparing values for five sets, increasing the pitch from 200 μm to 400 μm doubles the dynamic noise.

Alignment of power/ground bumps and TSVs is an important factor in power-supply noise and IR-drop. Figure 50 shows the percentage change in IR-drop as a function of the offset between TSV stacks and bumps. In the case where offset is nonzero, all the current traveling through the TSVs must detour through the relatively small wires in the power distribution grid. The graph shows the case where TSVs are $40 \times 40 \mu m$ in dimension. The

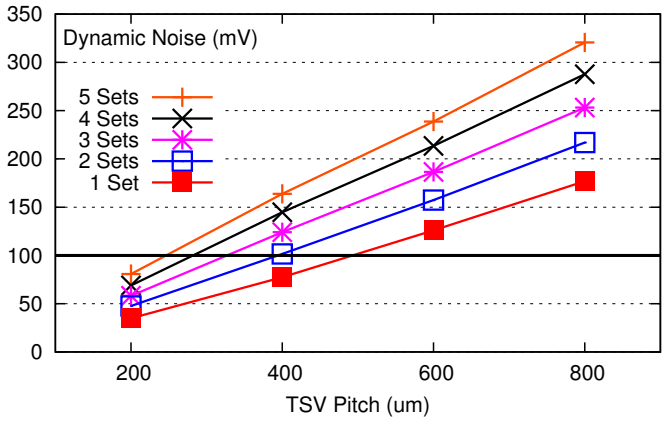


Figure 48. The dynamic noise as a function of TSV/bump pitch; the baseline case is 400 μm.

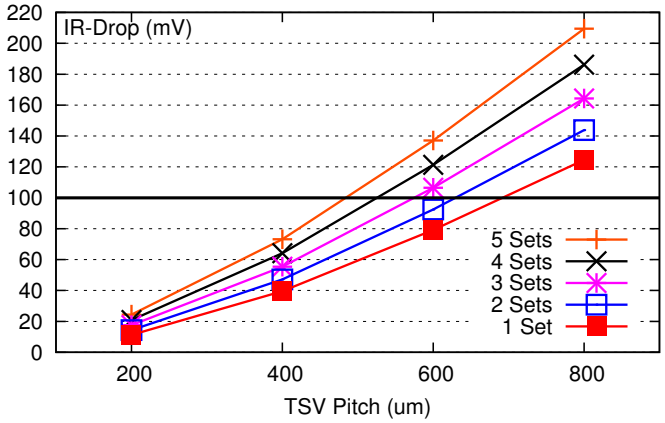


Figure 49. The IR drop as a function of TSV/bump pitch; the baseline case is 400 μm.

results show that misaligning TSVs from the bumps is a very bad idea; for the five-sets-stacked case the maximum IR-drop increases by 160% for maximum misalignment.

The majority of our results are for cases where the contact resistance between stacked TSVs is much smaller than the resistance of the TSVs themselves. This corresponds to TSVs that are manufactured as a single unit/pillar or are bonded very tightly. To examine the effects of this assumption, we performed experiments that vary the contact resistance. Figure 51 shows the IR-drop scaling for increasing contact resistance on a log scale. The graph shows that even $10m\Omega$ is too large of a contact resistance for large 3D stackings.

Figures 52 and 53 show the dynamic noise and IR drop, respectively, for various thin-wire sheet resistance values. This shows the noise sensitivity of 3D stackings to

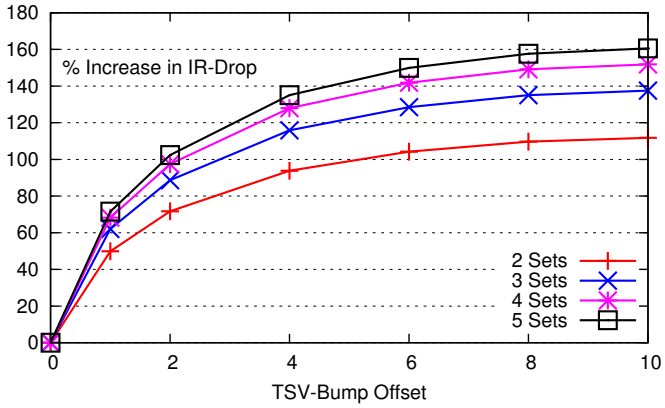


Figure 50. The effect of misalignment between TSVs and package-level bumps on IR-drop.

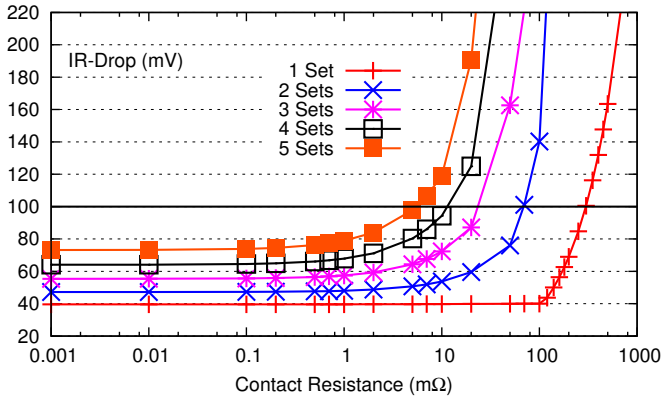


Figure 51. The effect of contact resistance on IR-drop. Note that the contact resistance is on a log scale.

distribution-grid wire geometry. The baseline case corresponds to copper wire geometry of 0.5μm thickness and 1μm width. Increases in wire width and thickness can increase the feasibility of higher stackings.

4.5.3 Other Organization Styles

This section focuses on differences between the core organization styles. Previous graphs have had roughly similar behavior between the organization styles. The cores-last and cores-first styles have similar responses, but the cores-last style has worse noise than the cores-first organization style. In general, the cores-spread organization style has lower dynamic noise than the other styles because of the large amount of decap provided by the memory tiers. Even though the decap density of the memory tiers is much lower than the processor tiers, the large number of memory tiers makes the total decap value high.

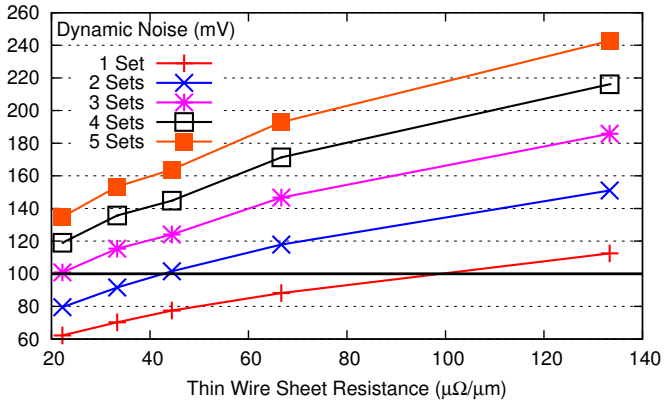


Figure 52. The effect of thin wire sheet resistance on dynamic noise. The baseline value is $67\mu\Omega/\mu m$.

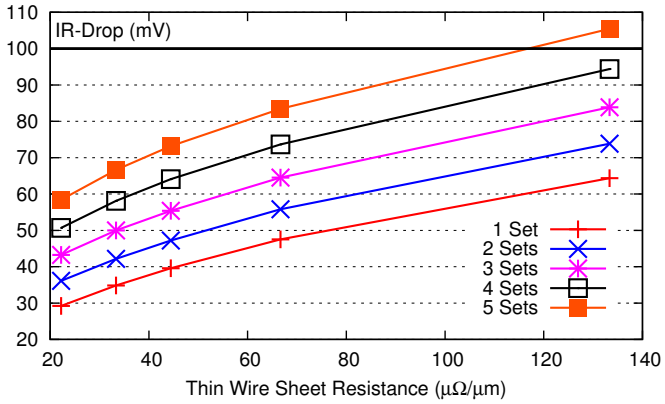


Figure 53. The effect of thin wire sheet resistance on IR-drop. The baseline value is $67\mu\Omega/\mu m$.

We begin by examining the effects of changing various parameters relating to the current demand and decap distribution. First, in Figure 54 we demonstrate the result of changing the rise-time of the processor tiers. In this experiment the rise-time for the memory-tiers was kept constant. The baseline rise-time of 5 *ns* is quite conservative, however, this should not impact the rest of our scaling studies. We observe that the rise-time benefits the cores-spread organization style significantly more than the cores-first style. This is a result of the much larger sensitivity to dynamic noise of the cores-spread style.

In Figure 55 we show the changes in dynamic noise as the decap density in the memory tiers is varied. As expected, the cores-spread organization shows a much higher sensitivity to memory decap density than the cores-first organization. For very low values of decap density, there is a crossover point where the cores-first organization performs better than the cores-spread organization because of the processor's closer proximity to the power-delivery

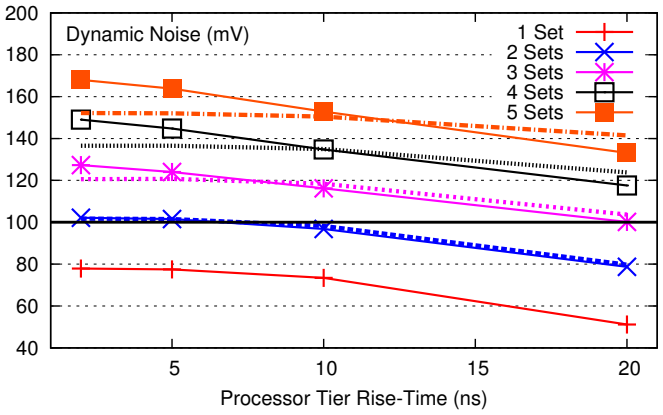


Figure 54. The dynamic noise as a function of processor-tier rise-time in ns. The baseline value is 5 ns. Dotted lines correspond to the cores-spread organization style.

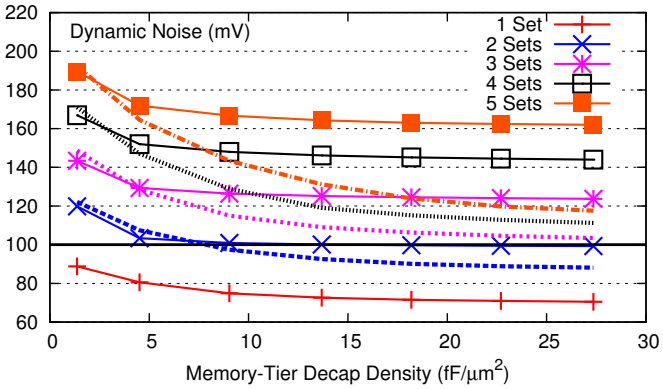


Figure 55. The dynamic noise as a function of memory tier decap density. The baseline case is approximately 8 $fF/\mu m^2$. Dotted lines correspond to the cores-spread organization style.

bumps.

Next, Figure 56 shows the dynamic-noise sensitivity to the processor-tier decap density. This study more clearly shows a crossover point. However, the amount of decap required to match the results for the high memory-tier-decap-density cores-spread experiments rises as the number of sets increases. Overall, a comparison between Figures 55 and 56 shows that increasing memory-tier decap with the cores-spread organization style is much more effective due to the fact that there are so many more memory-tiers than processor-tiers. Conversely, for the cores-first organization the processor-tier decap density is more important.

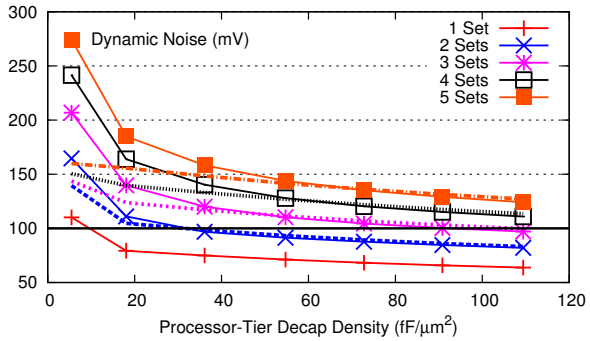


Figure 56. The dynamic noise as a function of processor tier decap density. The baseline case is approximately 37 $fF/\mu m^2$. Dotted lines correspond to the cores-spread organization style.

4.5.4 Temperature Scaling Results

Figure 57 shows the maximum temperature scaling among the three core-organization

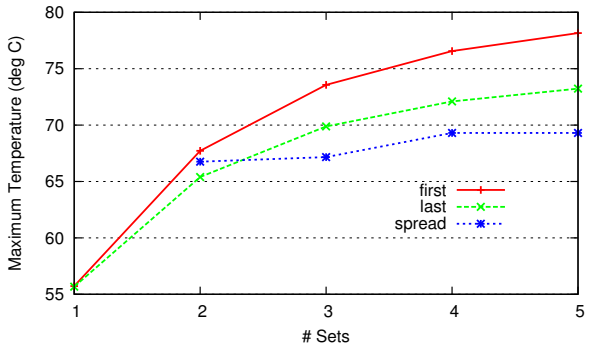


Figure 57. The temperature scaling of the three core-organization styles. The vertical axis denotes the maximum temperature observed within each chip stack using micro-fluidic heatsinks.

styles. The horizontal axis denotes number of sets stacked together. The vertical axis denotes the maximum temperature observed within the chip stack. The cores-first organization style has the worst temperature-scaling behavior, but the value is still below the thermal packaging limits of modern technology. The cores-spread organization style has the best scaling behavior among the three styles.

Figure 58 shows the per-tier temperature for the cores-first organization style. The horizontal axis denotes the tier number and the vertical axis denotes the maximum temperature observed on that tier. The memory tiers have no micro-fluidics so the temperature of those tiers increases with stacking. It is apparent that with higher memory-tier power consumption they may dominate the temperature of the processor tiers. In this case micro-fluidic

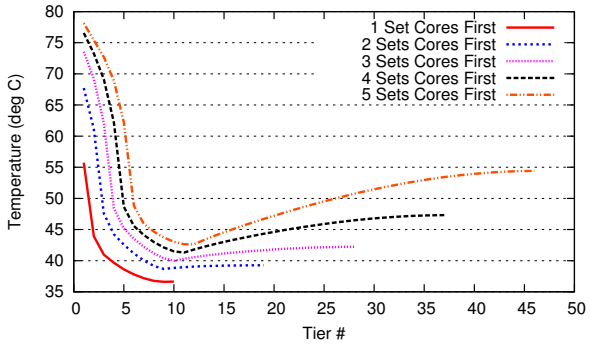


Figure 58. The per-tier temperature of the cores-first organizational style. The vertical axis denotes the maximum temperature observed within each tier.

channels could be added to some memory tiers to lower their temperature. Experiments show that a memory-tier power dissipation of approximately 1.3 watts-per-tier causes the maximum memory-tier temperature to match that of the cores for a five-set cores-first stacking. This value is more than double the value assumed for the rest of our experiments.

The per-tier temperature scaling of the cores-spread organization style is shown in Figure 59. The humps show the location of the processor tier within the stack. The maximum

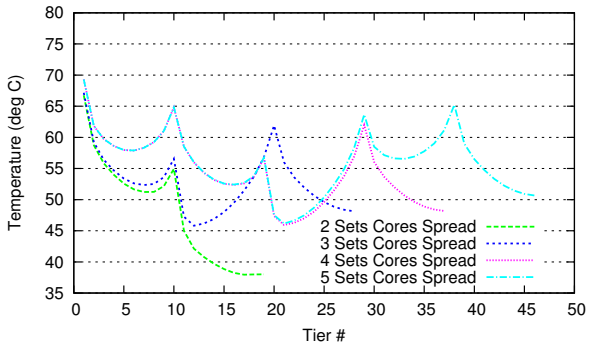


Figure 59. The per-tier temperature of the cores-spread organizational style. The vertical axis denotes the maximum temperature observed within each tier.

temperature is always observed to be at the tier closest to the bumps and farthest from the air-cooled heatsink. This is explained by the distance between neighboring processor tiers in the cores-spread organization style. This style has the lowest volumetric power density of the three options.

Figure 60 shows the per-tier temperature for the cores-last organization style. As expected, the scaling behavior is similar to that of the cores-first organization style; it also

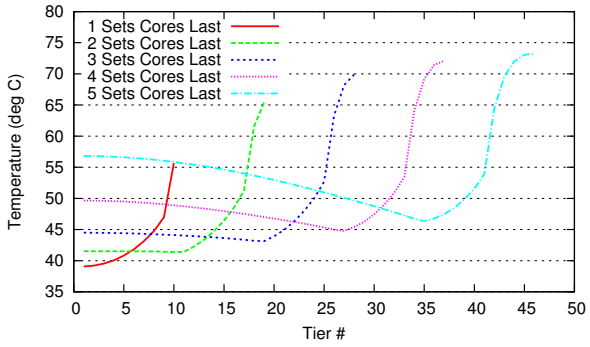


Figure 60. The per-tier temperature of the cores-last organizational style. The vertical axis denotes the maximum temperature observed within each tier.

exhibits the same memory-tier bump as the cores-first style.

4.5.5 TSV Topology

Figures 61 and 62 show the dynamic noise and IR-drop results for the various TSV topolo-

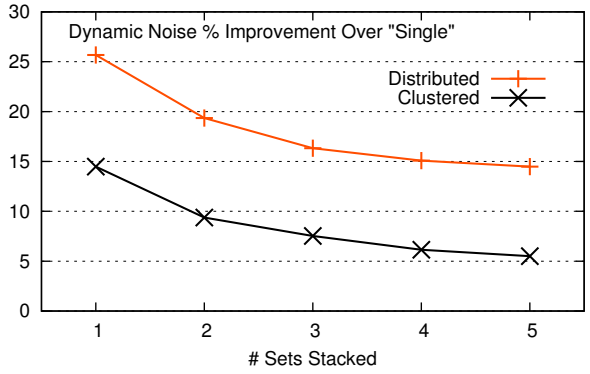


Figure 61. The % improvement of the distributed and clustered TSV topologies over the single TSV topology on dynamic noise. The horizontal axis denotes the number of sets of our scalable prototype stacked together.

gies discussed in Section 4.1. The reduced effective inductance for both the clustered and distributed topologies are included. The single topology uses the baseline TSV dimensions of $40 \times 40 \mu m$. The graphs show that the use of the distributed topology in conjunction with the large number of low power tiers represented by the DRAM and memory controller tiers can effectively lower the noise values of the processor tiers. In the best case, the distributed topology reduces dynamic noise for the one set stacked case by 25.6% and IR-drop by 51.1%. Examination of Figure 62 shows that the higher C4-to-TSV resistance for the clustered topology can cause an *increase* in DC IR-drop for higher numbers of tiers stacked

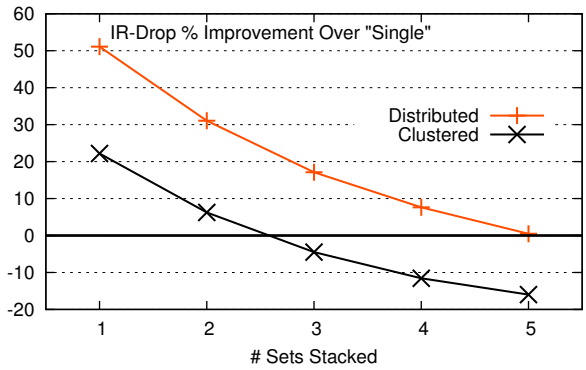


Figure 62. The % improvement of the distributed and clustered TSV topologies over the single TSV topology on IR-drop. The horizontal axis denotes the number of sets of our scalable prototype stacked together.

together. However, only with more than 50 tiers stacked together is the IR-drop for the distributed TSV topology likely to be higher than that of the single TSV topology.

4.5.6 Staggered Turn-On Policy

Given the worst-case nature of the previous results and the fact that the dynamic noise is above the 10% noise threshold for stackings of two or more sets on our default power grid dimensions, we examine adding a staggered turn-on policy in this section. Our staggered turn-on policy prevents processor tiers from coming out of the sleep state at the same time.

Figure 63 shows the dynamic noise results for adding various delays between the sleep

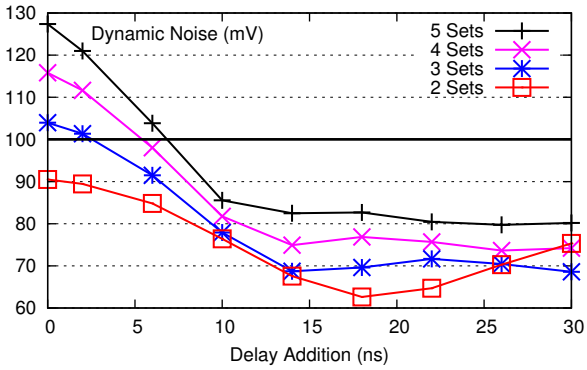


Figure 63. The effect of adding a staggered turn-on policy to the processor tiers on dynamic noise. The maximum dynamic noise reduction is over 37%. For all stacking cases the dynamic noise can be lowered below the 10% noise margin.

transitions of successive processor tiers. The zero-added-delay points show the default case, where all tiers activate simultaneously. The graph shows that for the five-sets case,

adding a 30*ns* delay reduces dynamic noise by over 37%. This turn-on policy reduces the noise below the 10% noise threshold. The performance impact of our turn-on policy is essentially negligible. Sleep transitions occur at the frequency of the operating system's scheduling interval. This interval is generally on the order of milliseconds, while our turn-on policy delays a processor from becoming active for at most 120 nanoseconds in the five-sets case, an overhead of only 0.01%.

4.5.7 Best Combined TSV Topology and Turn-On Policy

The addition of a turn-on policy is independent of the TSV topology chosen when designing the power distribution network. Figure 64 shows the improvement of the best possible design over the baseline case for our scalable system by combining these two techniques.

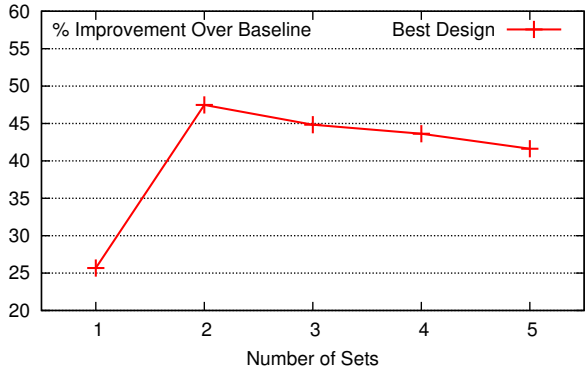


Figure 64. The percentage improvement over the baseline of a design that combines the distributed TSV topology with a noise-limiting turn-on policy. All of the data points represent noise values below the 10% noise margin.

The best improvement is seen for the two-sets-stacked case with a 48.5% decrease in dynamic noise. The large improvement seen when moving to the two-sets case is a result of the fact that the dynamic noise is already quite low for one-set-stacked case. The data point representing one set stacked essentially has no turn-on policy and therefore is simply the same as the distributed TSV topology result. The difference between the two-sets and one-set stacked case can be seen by examining the voltage swing curves in Figure 65. For the two-sets case, the first set is swinging back upwards when the second set begins to turn on, which reduces the maximum voltage swing. This behavior may be difficult to replicate

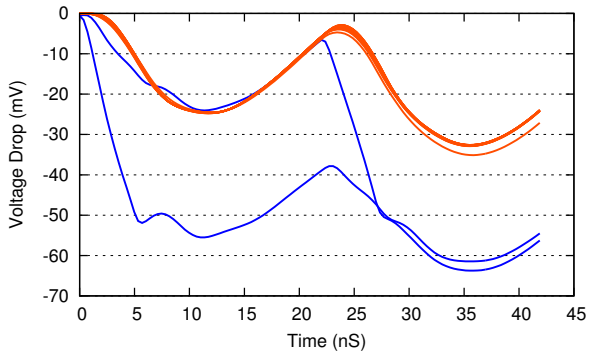


Figure 65. An example of voltage drop as a function of time with the turn-on policy. Each line represents the noisiest point on each tier. The two processor tiers are colored blue.

in more realistic situations.

4.6 Summary

This chapter presents scaling studies related to through-silicon via (TSV) parasitics and power-distribution networks for large-scale 3D systems. This chapter also presents two basic design techniques for reducing power-supply noise. Our basic model is composed of a scalable prototype system that contains nine tiers, including four high-performance processor cores and eight gigabytes of DRAM. Consideration of micro-fluidic heatsinks is included as it applies to the design of the power-supply grid.

Inductance of TSVs is a major factor that impacts power-supply integrity in large-scale 3D systems. Our results show that ignoring TSV inductance leads to 14.8% underestimation of dynamic power-supply noise in the largest system we examine. Additionally, TSV and package-level power-supply-bump pitch are one of the most critical factors in reducing power-supply noise and IR-drop in these devices. Contact-resistance between TSVs may become a limiting factor. However, this is largely dependent on manufacturing technologies. Overall, our studies indicate that stackings with up to 46 tiers are possible, given that sufficient care is given in the design of the power-distribution network. Additionally, a cores-spread tier organization style can generally outperform a cores-first organization style for dynamic noise. However, the relationship is reversed for IR-drop.

Our thermal scaling studies using micro-fluidic heatsinks also indicate that this type of

large-scale integration is feasible. Temperature scaling is best for the cores-spread organization style, but stackings with up to five sets in the other organization styles remain within current packaging thermal limitations.

Implementing a distributed arrangement of small (signal-sized) TSVs results in the lowest TSV inductance, and, when used in a system with both high-power and low-power tiers, can result in the lowest dynamic noise and IR-drop of any of the alternatives examined. Also, the addition of a staggered turn-on policy for neighboring cores can significantly reduce the noise generated by sleep transitions. Combining distributed TSVs with a staggered turn-on policy reduces dynamic noise in our simulations by up to 48.5%, and IR-drop by up to 51.1%, compared to the baseline case.

CHAPTER 5

3D POWER-SUPPLY-NETWORK DESIGN

3D stacking of ICs has generated increasing interest from the VLSI community in recent years. The many potential benefits of 3D integration include reduced power consumption from off-chip communication, reduced wirelength and delay, and lower-cost process integration. However, there are many challenges involved in the design of 3D ICs that have not been met. Increased volumetric power density combined with increased thermal resistance between the lower layers and the heatsink imply increased operating temperatures and an associated reduction in reliability. Smaller footprints combined with larger package-level system power imply increased power delivery problems. Solutions to all of these problems are the subject of ongoing work in both academia and industry. In this work we provide a layout-level examination of the design of 3D power delivery networks, and demonstrate that the unique environment of 3D ICs can have a dramatic effect on IR-drop and dynamic noise in these networks.

IR-drop (sometimes referred to as ground-bounce) is the resistive voltage drop in power and ground distribution networks caused by the dynamic and leakage power of ICs. IR-drop causes many problems in modern microprocessor and ASIC designs, and helped bring about the end of the frequency scaling era. As device scaling continues, lower and lower supply voltages are increasing total current and reducing power-supply-noise margins even further. These issues are causing an increasing percentage of available routing resources to be dedicated to power-supply distribution in high-performance designs, which can add significantly to congestion problems and reduce the amount of functionality that can be packed into a unit area.

Dynamic supply noise (sometimes referred to as dI/dt noise or simultaneous-switching noise) is transient voltage instability in power and ground distribution networks caused by the interaction of the capacitance and inductance of those distribution networks with

variable switching activity of transistors. Dynamic noise causes problems with timing closure and device reliability, because lower supply voltages cause transistors to switch more slowly. Decoupling capacitance (decap) is typically added to the power-distribution network to mitigate the effects of dynamic noise, however, large amounts of decap can cause significant increases in leakage power. Modern designs require large amounts of decap to meet supply-noise constraints. Techniques that reduce decap requirements are valuable additions to an IC designer's toolkit.

Many researchers have proposed optimization schemes for traditional IC power network design. Previous work on 3D power-delivery networks has largely assumed a straightforward extension of 2D power-delivery network design. Huang *et al.* [69] presented a physical model of 3D power-distribution networks. In their model, power/ground through-silicon vias (TSVs) and power-supply C4 bumps are always aligned with one another. Jain *et al.* [70] extended the work of Gu *et al.* [71] by examining the use of multi-story power delivery in 3D ICs. In their approach, there are two power domains and the ground network of one domain is the power network of the other domain. Again, the TSVs and supply bumps are always assumed to be fully aligned, and they are divided among the three power distribution networks evenly. Yu *et al.* [72] demonstrated an optimization scheme for supply-bump assignment and via insertion that simultaneously considers both supply noise and temperature. They again assume that supply bumps are aligned with TSVs in every case.

The overall goal of this chapter is to explore power delivery in 3D ICs and how it differs from traditional designs. Compared to prior efforts, we demonstrate the benefits of reexamining the unique capabilities of TSVs relative to package-level bumps. We also perform our analysis using layout-level designs and validate our modeling results using commercial-grade sign-off IR-drop analysis software. The major contributions of this work are as follows:

- We present the first layout-level analysis of 3D power distribution networks that is

validated using commercial tools.

- We demonstrate the potential IR-drop and dynamic noise benefits of spreading power and ground distribution TSVs away from the power and ground supply bumps in designs with non-uniform power dissipation.

- We examine scaling trends in 3D power-distribution networks using this framework to demonstrate future potential for increased 3D stacking on an envisioned 1000-core system.

- We analyze several modifications of power-distribution network design unique to 3D systems, and show their effects on IR-drop and dynamic noise.

5.1 3D and Flip-Chip Power Networks

High performance 3D systems will generally use flip-chip-style packaging to increase off-chip interconnect density and reduce parasitics. Flip-chip power distribution systems are commonly laid out as grids. High-level metal layers are reserved for laying out a coarse-grained grid with large wires that connects a regular array of power and ground C4 bumps. A fine-grained mesh provides local distribution and connects to lower-level-metal power rings or standard-cell row distribution wiring. Most commercial products today have C4 bump pitches around 100 to 200μm, however, researchers have demonstrated micro-bumps with pitches below 10μm.

For 3D systems, the TSVs will fill the role of the C4 bumps for intermediate tiers. Each tier will contain its own power distribution network. Figure 66 shows the general topology of a 3D power-distribution network. The vertical resistance between adjacent tiers should be close to that of the C4 bumps to maintain reliable power and ground voltages in large-scale 3D systems. The resistance of individual C4 bumps is on the order of 5$m\Omega$. Additionally, TSVs should be smaller than the C4 bumps, or large amounts (25% or more) of die area will become unusable.

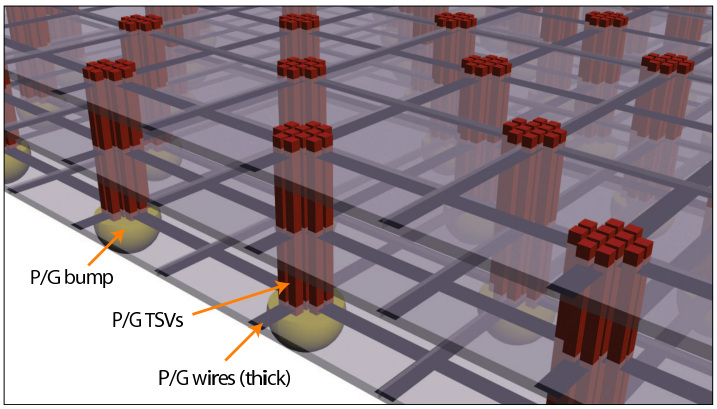


Figure 66. Bumps, TSVs, and wires in a 3D P/G network

TSVs can be manufactured in many different sizes. Diameters of less than $1\mu m$ have been shown in the literature. Power and ground TSVs should be large to have low resistance, but signal TSVs should be small to increase interconnect density and reduce parasitic capacitance. Manufacturing multiple TSV sizes on a single die would increase cost and reduce yield. Therefore, it will likely be necessary to use a single TSV size for both power distribution and signal wiring.

In this work it is assumed that only one TSV size is available, and is optimized for signals. There are several potential combinations of TSV distribution that could be used to deliver power. Figure 67 shows two of the basic choices investigated thoroughly in this chapter.

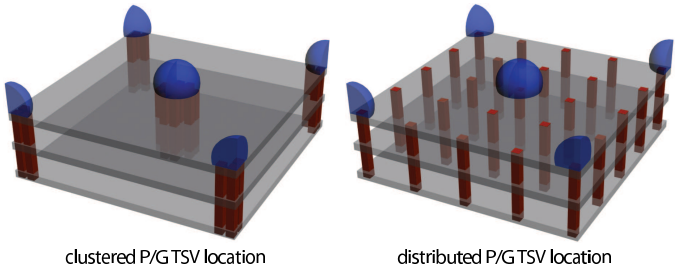


Figure 67. Two TSV topologies for power distribution in a single tile of the distribution network. C4 bumps are shown in blue and P/G TSVs in red. The combined resistance of all TSVs in each topology is equal.

- **clustered topology**: multiple small TSVs are clustered over the C4 pads for both power and ground distribution.

- **distributed topology**: multiple small TSVs are distributed evenly throughout the die for both power and ground distribution.

For both of the topologies the combined resistance of all the TSVs is assumed to be the same. The figure depicts TSV topologies for a single tile in the power/ground network. This tile is mirrored and replicated all over the chip.

5.1.1 TSV Topology IR-Drop Comparison

A simple 1-D example demonstrating the difference between the clustered and distributed TSV topologies is shown in Figure 68. The ground network of a two-tier system is modeled using current sources and resistances. The lower tier in (a) and (b) has twice the power

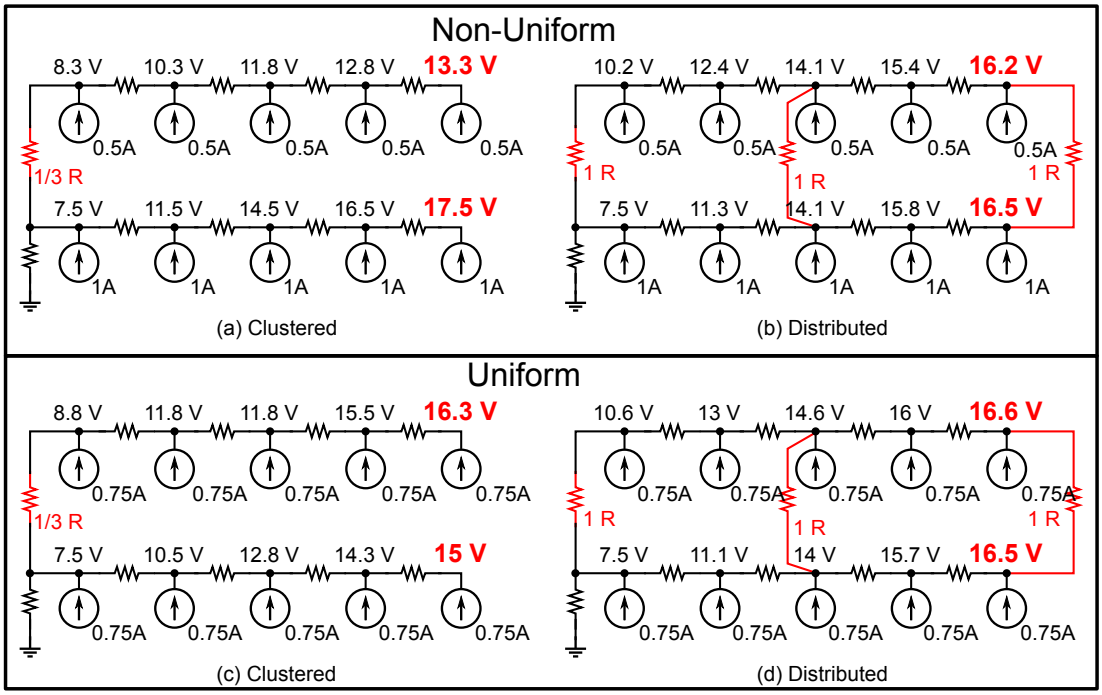


Figure 68. A simple 1-D example that demonstrates the power-supply-noise improvement encountered when using the distributed TSV topology. Non-uniform per-tier power dissipation is shown in (a) and (b). A uniform per-tier power dissipation version with the same *total* power dissipation is shown in (c) and (d). All resistance values are $R = 1\Omega$, except where noted in red.

dissipation of the upper tier. The upper tier is connected to the lower tier with three TSVs either clustered together on the left side ((a) and (c)), or distributed throughout the design ((b) and (d)). The figure shows the voltage at every node of the network for all cases. In the

non-uniform circuits, the maximum voltage is 17.5 V for the clustered case and 16.5 V for the distributed case, an improvement of 5.7%. This example demonstrates the basic reason why the distributed TSV topology generally results in better power-distribution network performance. In the clustered case the difference between the maximum per-tier voltages is $17.5 - 13.3 = 4.2V$. This represents the "slack" in the upper tier. For the distributed topology, the slack in the upper tier is much lower, around $0.3V$, and the maximum drop on the lower tier is also lower than for the circuit with the clustered topology.

Figure 68 (c) and (d) show the same comparison for the case with *uniform* per-tier power dissipation. The *total* power dissipation is also the same as in the non-uniform case. In this case it can be seen that the clustered topology results in lower maximum IR-drop than the distributed topology. This occurs because the voltage difference between the two tiers is so small. However, the difference between the two TSV topologies is also relatively small.

5.1.2 TSV Topology Area Overhead Comparison

The difference between the coefficients of thermal expansion (CTE) of silicon and TSV conductors causes thermal stress in the silicon die. This thermally-induced stress can affect device performance. TSV manufacturing processes may also negatively impact nearby device performance and manufacture. For these reasons, gates and transistors are generally placed outside of a keep-out region (KOR) around the TSVs. Figure 69 shows a group of TSVs with the KOR highlighted and also defines several dimensions associated with the KOR. The figure defines K as the distance of the edge of the KOR from the TSV, T as the dimension of the TSV, and S as the TSV-to-TSV space. The area taken up by an $n \times m$ array of TSVs in the clustered topology is then:

$$A_{clustered} = (2K + (n-1) \cdot S + n \cdot T) \times (2K + (m-1) \cdot S + m \cdot T). \quad (20)$$

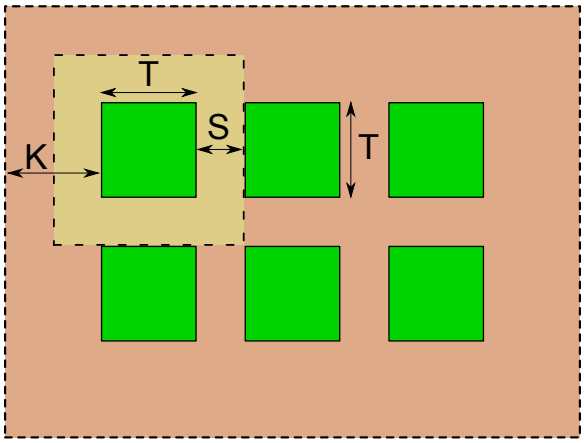


Figure 69. An illustration of the keep-out region (KOR) around a group of TSVs. The distance of the edge of the KOR from the TSV is defined as K, the dimension of the TSV is T, and S is the TSV-to-TSV space.

Additionally, the area taken up by an $n \times m$ array of TSVs in the distributed topology is then:

$$A_{distributed} = n \cdot m \cdot (2K + T)^2. \tag{21}$$

It is obvious that the distributed topology will occupy more silicon area than the clustered topology when S is small. Figure 70 shows the area overhead for a 5×5 array of TSVs

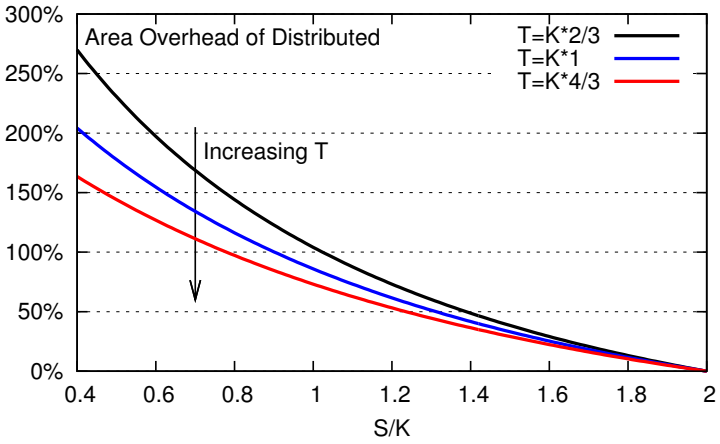


Figure 70. The area overhead for a 5×5 array of TSVs in the distributed topology compared to the clustered topology. The ratio between S and K is varied on the independent axis. Data for several different values of T are shown.

in the distributed topology compared to the clustered topology. The ratio between S and K is varied on the independent axis. Data for several different values of T are shown. Both topologies occupy the same area, and the overhead is zero, when $S = 2K$. The area

overhead is 86% when $S = K = T$ for $n = m = 5$. It should be noted that for the prototype layout presented in the next section, the total area occupied by all of the $6 \times 6\mu m$ TSVs is less than 5% of the total area. Additionally, the power-supply TSVs are generally located under the power and ground stripes, which already constrain placement and routing.

We compared several different routing solutions using the clustered and distributed topologies for the layouts described in the next section. The distributed topology resulted in slightly more than a 0.05% increase in total wirelength for the core layer, and no difference for the memory layers. In designs with more significant routing congestion, this difference may become important. In this case power distribution TSVs may need to be moved to reduce such congestion, which could impact the effectiveness of the distributed TSV topology.

5.2 Prototype Layout

The prototype layout used in our simulations is based on a design targeted at demonstrating extreme memory bandwidth using 3D interconnects. Our design is a many-core processor composed of an array of simple cores connected with a nearest-neighbor communication mesh. Each core has eight banks of dedicated SRAM directly stacked above it in two separate tiers. Each core tier contains a 10×10 array of cores. One grouping of one core tier and two SRAM tiers is defined to be one "set" of our scalable prototype layout. We envision stacking 10 sets together to form a 1000-core processor. The full 1000-core processor is shown in Figure 71.

The layouts used in our experiments were designed using a 130-nm standard cell library from Global Foundries. For the physical design, we used Cadence's SOC Encounter automated place and route tool. The layouts for a single core and a single memory tile are shown in Figure 72. We also highlight the areas in the layout reserved for ground TSV connections. The distribution of connection points is irregular due to the constraints of the layout, especially the locations of the hard memory macros. For the distributed TSV

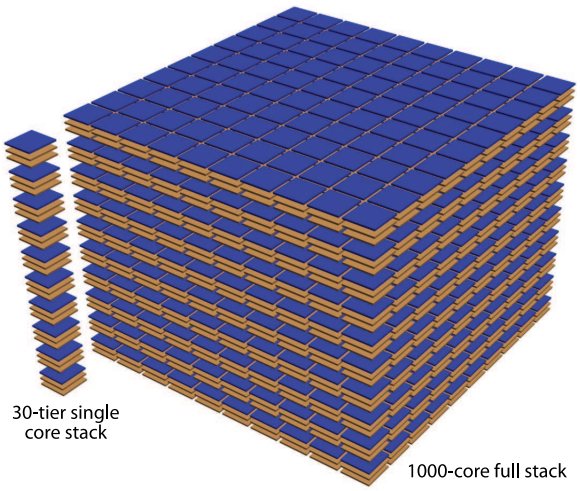


Figure 71. The 1000-core processor that is targeted in our simulations. Our sign-off noise simulation covers the 30-tier single core stack.

topology, TSVs are located at all of the potential locations. In the clustered TSV topology, all of the TSVs are grouped into the center position, over the C4 bump. The power TSV locations are similarly distributed in an offset fashion; the main difference is that the power C4 bumps are at the corners of the core, while the ground C4 bump is in the center. Each location is capable of accepting a $6\mu m$ diameter via-first TSV, while the locations over the C4 bumps (the center and near the corners) are capable of accepting 25 or more of these TSVs.

The single-core and single-tile layouts are both $560\mu m$ square. The core-to-core and tile-to-tile pitch is $590\mu m$ to accommodate the inter-core logic and communication, as well as the power distribution from the C4 bumps. The full 100-core and 100-tile layers are approximately $6mm$ square. Each core tile has $21.9pF$ of decoupling capacitance ($219pF$ per tier), and each memory tile has $21.7pF$ of decoupling capacitance ($217pF$ per tier). The maximum total power dissipation per set (one core tier plus two memory tiers) is approximately $13.2W$, the 1000-core system then has a total power dissipation of $132W$. Figure 73 shows the power map for a single core. The power dissipation of this design is not extreme, however, the high volumetric power density could be a problem for traditional heatsinks. For this case, micro-fluidic channels [101, 116] have been shown to be an effective method

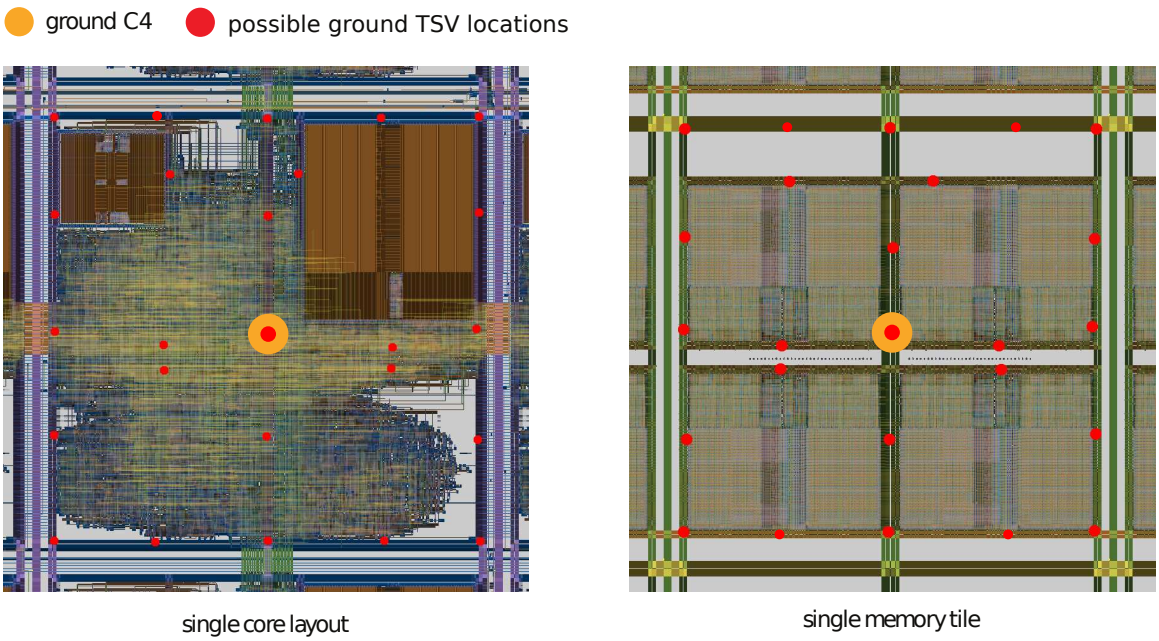


Figure 72. Layout of a single core and single memory tile from our 1000-core processor. The possible ground distribution TSV locations are highlighted in red. The ground C4 bump in the center of the core is indicated. The power C4 bumps are near the corners of the core.

for cooling large-scale 3D chip stacks, as discussed in Chapter 4.

5.3 3D IR-Drop Analysis

5.3.1 Methodology

Layout-level IR-drop values are computed by performing power consumption simulations, either statistical or simulation-driven, to obtain gate- and module-level power consumption values. The consumption values are then divided by the nominal supply voltage, in our case $1.5V$, to obtain gate- and module-level current consumption values. Next, parasitic extraction is performed on the layout to obtain a SPICE netlist that models the power distribution network. Our experiments were performed using Cadence's QRC transistor-level extraction tool. The current consumption values are then connected to the nodes representing the corresponding transistors belonging to the appropriate gates and modules. For traditional 2D ICs the netlist is then simulated using a power network simulator, in our case Cadence's UltraSim. Figure 74 shows the analysis flow 2D netlists. In 3D designs, the previously described steps are performed once for each type of tier (core, memory, etc.). The tier-type

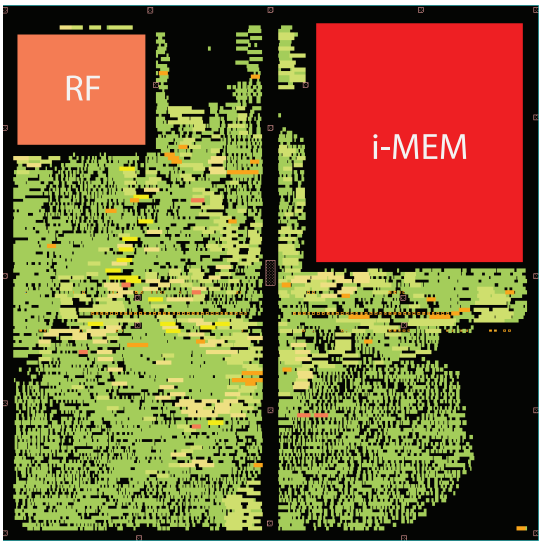


Figure 73. The power map for one core of our processor. The maximum total power consumption per core is $65.5mW$**.**

SPICE models are replicated for each instance of that tier-type and then connected using a resistive TSV model.

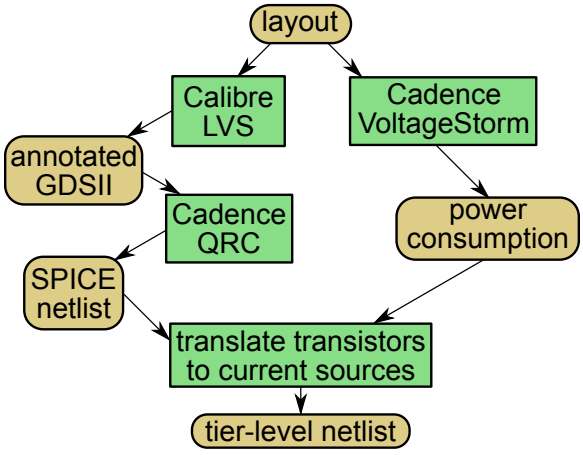


Figure 74. The analysis flow used to obtain the tier-level netlist for IR-drop analysis. This flow is performed multiple times for each tier type, then the netlists are connected together with a TSV model for 3D analysis.

Simulation of power-distribution networks is a generally difficult problem for traditional ICs. These networks can contain tens of millions of nodes. 3D stacking exacerbates the problem even further. Given the extreme regularity of the prototype design that is examined in this work, we mitigate some of the extreme memory and execution-time requirements of power-network simulation by only simulating an area containing a single

core and the tiers directly above it, as shown in Figure 71. We stress that our design is extremely regular and so this reduction should only impact the accuracy of our analysis in a minor way. Simulations indicate that this introduces approximately 3% error in our results. However, the error is systematic in nature, and should not affect the results of our scaling studies.

5.3.2 Validation

To validate the IR-drop analysis flow described above, we compare the results for a 2D layout to Cadence's VoltageStorm sign-off power-noise analysis tool. The results of our analysis flow are within 4% of the values reported by VoltageStorm. We were also able to create a method for using VoltageStorm for to perform 3D analysis for two-tier stacks.

First, we create an ICT file, a process technology description file, that contains a description of all of the metal layers in two tiers. The metal and dielectric layers are renamed so that the tier number is embedded in the name. For example, "METAL1" becomes "METAL1_1" and "METAL1_2." Then, a techfile is created using Cadence's TechGen based on the new ICT file. Next, we modify the LEF files provided by the foundry that describe the technology, standard cells, and macros. The DEF and instance power files for the designs of each tier are also modified in the same way. Each file is essentially duplicated so that there is one version for the first tier and one version for the second tier. The modifications basically amount to renaming the objects and metal layers in the same way that the ICT file is modified. To include detailed analysis of the macro blocks we also modify their GDSII files. We first convert the GDSII to GDT, an ascii-version of the binary GDSII data. Then we map all of the GDSII layer numbers for the metal layers into a non-overlapping number space. The modified GDT is then converted back to GDSII. The XTC extraction tool is then given a GDSII layer map file that maps the appropriate layer numbers to the correct tier's metal layers for each macro.

Using the above method we were able to match the 3D IR-drop results from VoltageStorm within 4%. Figure 75 shows a depiction from Cadence's ViewICT tool of the

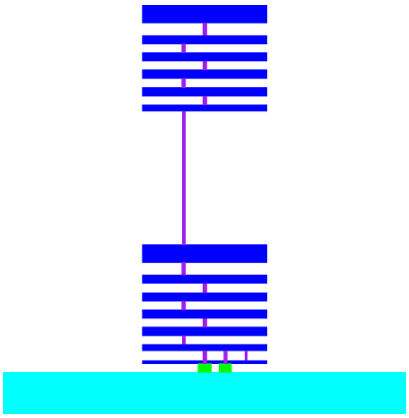


Figure 75. A depiction of the ICT file that contains metal layers for two tiers of a 3D stack. The ICT file is used to compile a techfile used for parasitic extraction by VoltageStorm for our 3D IR-drop verification flow.

modified ICT file containing metal layers in two dies. Note that the second die does not have a substrate layer. This is a limitation of the tool, due to the fact that it was not designed with 3D designs in mind. However, for power/ground network analysis the substrate can largely be ignored. For these experiments we created a face-to-back style 3D design, however, this technique is general enough to apply to face-to-face 3D designs as well.

Extending our 3D IR-drop analysis flow to 30 tiers is reasonable because IR-drop is the result of a linear system [39]. Both the per-tier current demand, I, and the per-tier resistance of the power-supply network, R, are unaffected by the number of tiers that are stacked together.

5.4 3D Dynamic Noise Analysis

Layout-level dynamic noise values are computed using the power consumption values and parasitic extracted networks obtained for IR-drop analysis with added decoupling capacitors. We create triangular current demand waveforms [39] (Figure 76) for each transistor such that the average power consumption matches the value obtained for IR-drop analysis. The triangular waveforms for each gate are delayed by a random amount such that the majority of them start near the beginning of the cycle. The random delays are distributed in a Gaussian fashion about zero and then the absolute value of the delay is used for the real

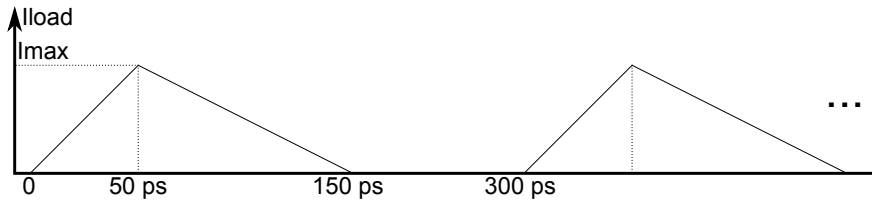


Figure 76. The current waveform used for each transistor for dynamic noise analysis. A random delay is added to the start of the waveform for each gate's transistors.

current waveform. Our dynamic noise results are obtained by performing transient simulation of a repeating pattern of current demand with a cycle time of $300ps$. After the voltage swings have stabilized, the peak of the swing is recorded as the noise value.

UltraSim's power network simulation engine does not handle large-scale transient simulation well, so we used a custom SPICE simulator based on Modified Nodal Analysis [111], which returns results within 2% of HSPICE. A step size of $1ps$ was used for our simulations. TSV inductance may also be an important contributor to dynamic noise. We modeled several sizes and arrangements of TSVs using Synopsys' inductance extractor Raphael [106]. The results of our TSV inductance simulations are shown in Table 10. The distributed TSV topology results in effective inductance values about two orders of magnitude smaller than the clustered TSV topology. The default TSV size used in most of our simulations is $6 \times 6 \times 20\mu m$.

Table 10. Effective inductance values in pH for power distribution TSVs. The TSV dimensions are in μm.

TSV Dimensions	Clustered	Distributed
$3 \times 3 \times 10\mu m$	$0.829pH$	$0.014pH$
$6 \times 6 \times 20\mu m$	$1.600pH$	$0.027pH$
$10 \times 10 \times 33\mu m$	$2.500pH$	$0.041pH$
$15 \times 15 \times 50\mu m$	$3.600pH$	$0.058pH$

5.5 Experimental Results

For our baseline analysis, we assume copper via-first TSVs with $6\mu m$ square diameter, $20\mu m$ depth, $35m\Omega$ resistance, and $1.6pH$ inductance. For simplicity, we present only the results for the ground distribution network. Simulations show that the power distribution

network has the same trends, only the location of the maximum IR-drop peak is shifted. In real designs the difference between the actual supply and ground voltages are what determine the performance of the gates. Given that we only simulate a single core and the tiers above it, we utilize a lumped package model for the C4 bumps. The C4 resistance and inductance in our simulations is $5m\Omega$ and $200pH$, respectively. Each of the memory tiers in our simulations consume about $0.7\times$ the power value of the core tiers, so the term "low-power tier" is somewhat relative.

5.5.1 Power-Supply-Noise Comparison: Clustered vs. Distributed

Figure 77 shows IR-drop and dynamic noise results comparing a 2D design with just cores

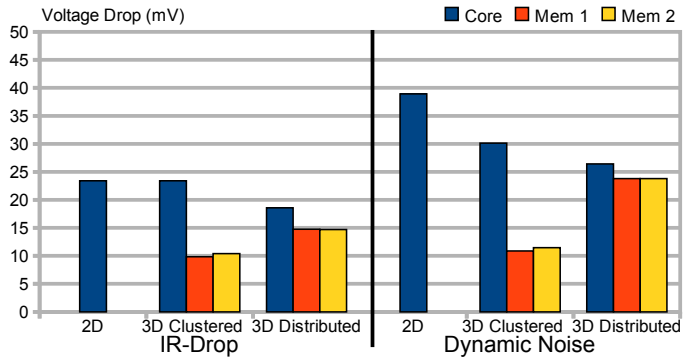


Figure 77. The per-tier IR-drop and dynamic noise results for a 2D design composed of cores only, a 3D design using the clustered TSV topology, and a 3D design with the distributed TSV topology. Both 3D designs consist of three stacked tiers, one core and two memories (one set of the scalable prototype).

to 3D designs using one set of our scalable prototype with both the clustered and distributed TSV topologies. The 3D design with the clustered TSV topology results in the same amount of IR-drop as the 2D design, but lower dynamic noise than the 2D design. The dynamic noise improvement is caused by the increase in on-chip decap present in the memory tiers. The 3D design with the distributed TSV topology results in the lowest IR-drop and dynamic noise of all three cases shown, for the reasons discussed in Section 5.1. The distributed TSV topology improves IR-drop by 21% and dynamic noise by 32% over the 2D system, even though the 3D system consumes more total power.

Figure 78 shows the effect on IR-drop of stacking more sets of the scalable prototype

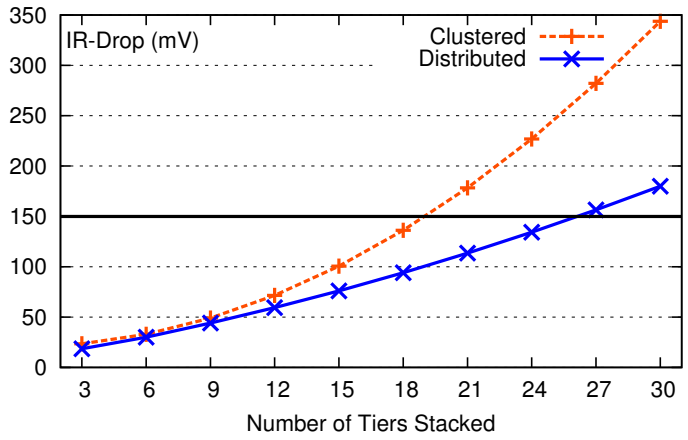


Figure 78. The change in IR-drop as more sets of the scalable prototype layout are added. The line at $150mV$ represents a 10% noise margin.

together. The distributed TSV topology provides a much lower IR-drop value as the number of sets stacked together becomes large. The distributed topology also allows up to six more tiers to be stacked together before crossing the 10% noise margin of $150mV$ compared to the clustered topology. The basic reason for this improvement in IR-drop is that the distributed TSV topology allows the tiers with the most IR-drop to accept current through the networks with lower IR-drop. The distributed topology effectively utilizes the "IR-drop slack" of the low-power tiers to lower the maximum system-level IR-drop.

For systems with fewer numbers of sets stacked in Figure 78, the clustered and distributed topologies result in very similar IR-drop values. Figure 80 shows the actual percentage improvement of the distributed topology over the clustered topology IR-drop for a few TSV site resistance values. These resistances are the resistance of each possible TSV location, called a TSV site. Figure 79 contains a representation of this arrangement. For the distributed topology, there are 25 such resistances spread throughout the core layout. For the clustered topology, there are 25 clustered at the center TSV location over the C4 pad. The resistances can represent multiple TSVs at each location in parallel. The results in Figure 80 show that TSV site resistance can have a significant impact on the relative IR-drop of the two topologies. However, for large numbers of sets stacked together, the distributed topology always eventually provides lower IR-drop than the clustered topology.

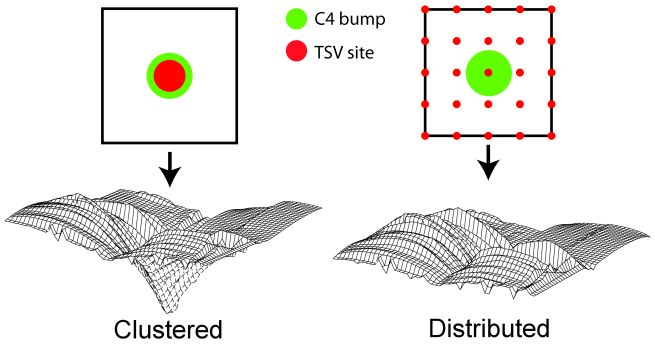


Figure 79. IR-drop meshes for a single core in the highest core tier of two sets of our prototype layout stacked together. The left graph shows the results for the clustered TSV topology and the right graph shows the results for the distributed TSV topology. Both meshes are plotted using the same scale.

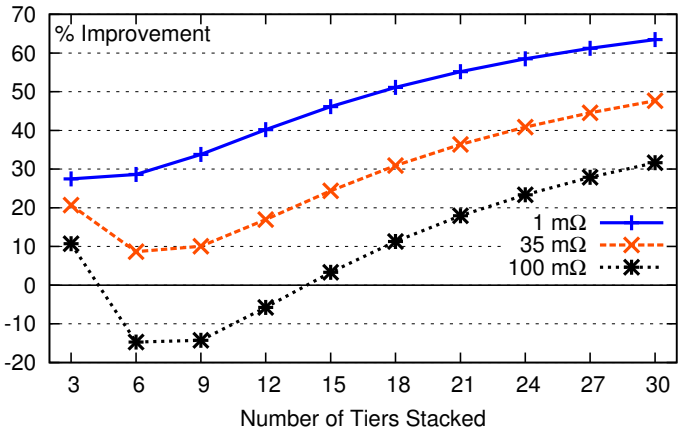


Figure 80. The IR-drop improvement of the distributed TSV topology over the clustered TSV topology as the number of tiers increases. Data for several values of TSV site resistance are shown.

Figure 81 shows the improvement of the distributed TSV topology over the clustered TSV topology for both IR-drop and dynamic noise. In general, the IR-drop and dynamic noise improvement show roughly the same trend. The IR-drop improvement is slightly higher in most cases.

Subsection 5.5.3 presents results relating to a wide range of TSV parasitic resistances. However, dynamic noise simulations are much more time-consuming than IR-drop simulations. Additionally, inductance is not a simple scalable quantity like resistance. For example, adding more TSVs to a TSV site to reduce the parasitic resistance will not reduce the parasitic inductance in a linear fashion. For these reasons this subsection presents dynamic noise results for a small set of TSV sizes using both TSV topologies. The TSV

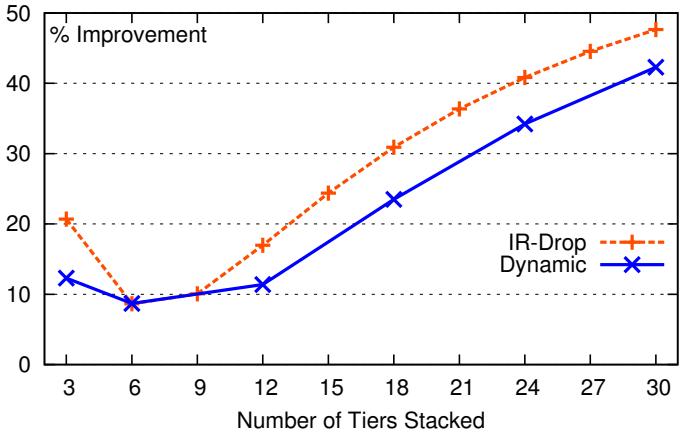


Figure 81. The improvement of the distributed TSV topology over the clustered TSV topology as the number of tiers stacked together increases. Both dynamic noise and IR-drop improvement are shown.

sizes and their inductance values are listed in Table 10. Figure 82 shows the dynamic noise

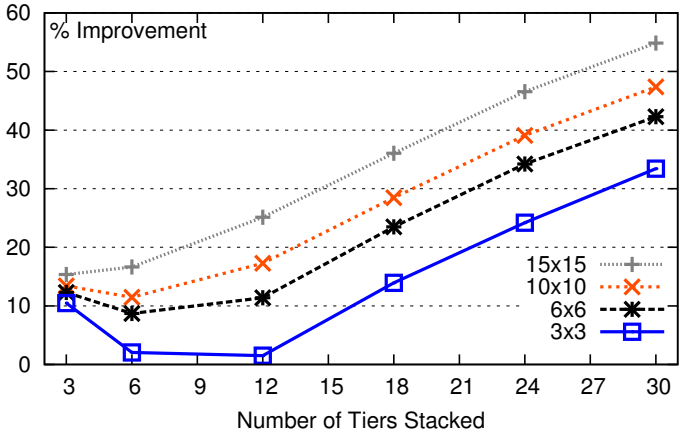


Figure 82. The dynamic noise improvement of the distributed TSV topology over the clustered TSV topology as the number of tiers increases. Data for several TSV sizes (and associated parasitics) are shown.

improvement of the distributed TSV topology over the clustered TSV topology for the four TSV sizes (and associated parasitics) examined. The dynamic noise shows trends very similar to those for IR-drop (Figure 80).

Figure 79 shows a 3D representation of the IR-drop over the surface of the core farthest from the C4 supply pads for a system with two sets of our scalable prototype layout stacked together. The figure shows that the clustered TSV topology produces an IR-drop map that has a much larger spread between the maximum and minimum values. The large dip in the center of the clustered TSV topology mesh indicates the position of the TSV connected

most directly to the ground network C4 bump for this core. The TSVs in the distributed topology help to pull down the IR-drop of the power distribution grid nodes that are farther from the C4 bump. The overall shape of the mesh demonstrates both that the TSVs are effective for lowering the maximum IR-drop, and that more TSVs should be even more effective for that purpose.

As a secondary consideration, IR-drop affects the switching speed of gates. In a distribution network with a much more uniform IR-drop, as in the case of the distributed TSV topology in Figure 79, the changes in gate delay would be more evenly matched between nearby gates. This smaller spatial variance in IR-drop should therefore create fewer problems for individual paths. In cases of excessive IR-drop, the entire system would slow as a whole, instead of individual paths causing significant timing issues.

To underscore the variation between the various tiers in the two TSV topologies, Figure 83 shows the maximum IR-drop values in each core tier of a system with ten sets of

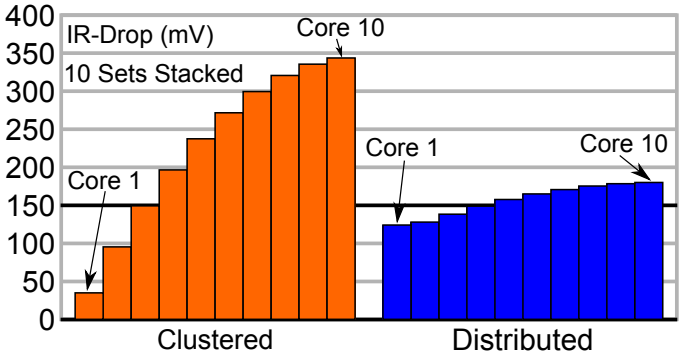


Figure 83. The maximum per-core-tier IR-drop for ten sets of our prototype layout stacked together. The spread in values of the clustered TSV topology is much larger than for the distributed TSV topology.

our scalable prototype stacked together. The TSV site resistance is set to the baseline case, $35m\Omega$. The difference between the maximum and minimum IR-drop in the system with the clustered TSV topology is more than $300mV$, while the difference is less than $60mV$ in the system with the distributed TSV topology. For the system with the clustered topology, the transistors on the lower tiers would be significantly faster than the transistors on the upper tiers.

5.5.2 Impact of Power Discrepancy Among Tiers

Next, we examine the effects of the power dissipation ratio between the memory tiers and the core tiers. As a reminder, the distributed TSV topology gains its IR-drop benefit from using the IR-drop slack of the low-power tiers to provide power to the high-power tiers. This implies that the total slack available (controlled by the power dissipation ratio) should effect the improvement of the distributed topology over the clustered topology. Figure 84 shows the effect of setting the ratio at 0.5 and 1.4, as well as the default 0.7. An interesting

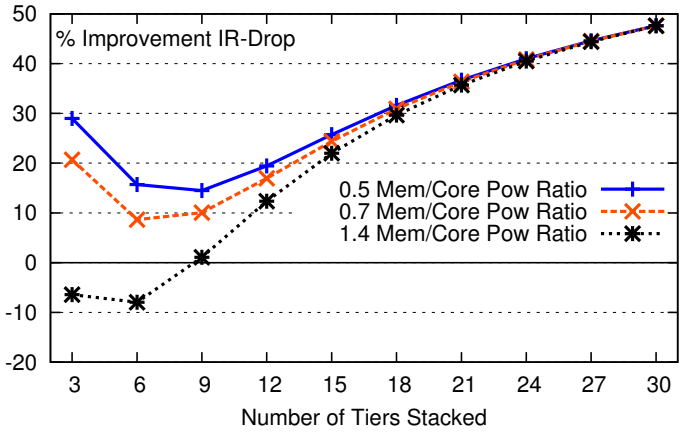


Figure 84. The IR-drop improvement of the distributed TSV topology over the clustered TSV topology as the number of tiers increases. The power dissipation ratio between the memory tiers and the core tiers is varied. The default ratio is 0.7 **and the TSV site resistance for all cases shown is** $35m\Omega$**.**

feature of the graph is that as the number of tiers stacked increases, the improvement of the distributed topology over the clustered topology becomes nearly identical for all cases. This indicates that the TSV resistance is more of a factor than the ratio of power dissipation between the low- and high-power tiers for these large-scale cases. For the case when power ratio is set to 0.5 there is extra slack available, so the distributed topology shows increased improvement. For the case when power ratio is set to 1.4 the core tiers are providing slack to the memory tiers.

Figure 85 shows the effect of varying the power dissipation ratio on the dynamic noise improvement of the distributed topology over the clustered topology. The dynamic noise improvement exhibits similar trends to the IR-drop improvement shown in Figure 84. As the number of tiers stacked increases the improvement of the distributed topology over the

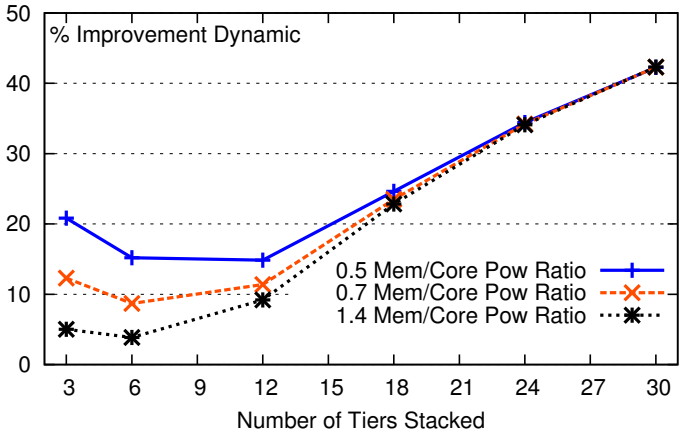


Figure 85. The dynamic noise improvement of the distributed TSV topology over the clustered TSV topology as the number of tiers increases. The power dissipation ratio between the memory tiers and the core tiers is varied. The default ratio is 0.7**.**

clustered topology becomes nearly identical for all power dissipation ratios. Again, this indicates that the TSV parasitics play a more important role in determining the improvement than the power dissipation ratio.

Figure 86 plots the effect on IR-drop improvement of the distributed topology over the

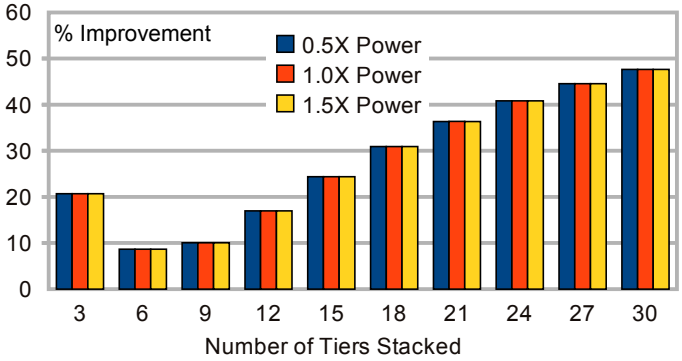


Figure 86. The improvement of the distributed TSV topology over the clustered TSV topology as the number of tiers increases. The total power dissipation is varied. The TSV site resistance for all cases shown is the baseline value, $35m\Omega$**. The improvements are identical regardless of total power dissipation.**

clustered topology for various multipliers of total power dissipation in the design. The graph shows that the total power dissipation has no effect on the improvement of the distributed TSV topology over the clustered TSV topology. This indicates that only the combination of the power distribution network topology involved and the ratios between the power dissipation of the various layers impact the IR-drop improvement of the distributed

TSV topology over the clustered TSV topology.

5.5.3 Impact of TSV Site Resistance on IR-Drop

Now we examine the effect of TSV site resistance on the IR-drop of the two topologies. The various values plotted could be created from longer or shorter TSV (and silicon) depth, different materials (tungsten, copper, etc.), varying interposer materials and contact resistances between stacked TSVs, or small arrays of TSVs in close proximity.

Figure 87 shows the per-tier maximum IR-drop trends resulting from scaling TSV site

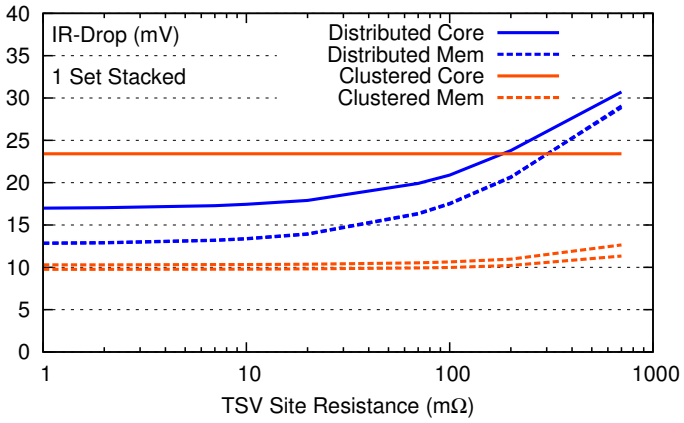


Figure 87. The effect of TSV site resistance on the maximum IR-drop of one set of our prototype layout. Note that TSV site resistance is on a log scale. The solid lines represent the core tiers and the dashed lines represent the memory tiers.

resistance in the case where one "set" of our scalable prototype layout is stacked together (one core tier and two memory tiers). The graph shows several trends. First, because only the TSV site resistance is being scaled, the package resistance remains constant, and the core tier in the clustered TSV topology maintains the same IR-drop irrespective of TSV site resistance. Second, the IR-drop scaling between the tiers of the distributed TSV topology shows a much stronger correlation, i.e., the maximum IR-drop of the various tiers in the system have values that are much closer together than for the clustered TSV topology. This indicates that the power networks of the neighboring tiers are tied together more strongly, and thus are able to support one another. Finally, while the distributed TSV topology has nearly 30% better IR-drop than the clustered topology for low TSV site resistance, the

crossover point between the two styles occurs at around 200$m\Omega$ TSV site resistance. For higher resistances, the distributed TSV topology begins to suffer from much higher IR-drop.

Increasing the number of sets stacked to two, we repeat the site resistance scaling simulations in Figure 88. To reduce visual clutter, we only show the results for the two core

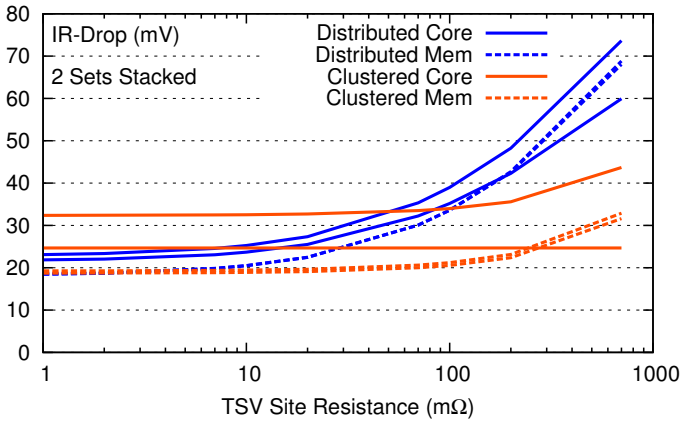


Figure 88. The effect of TSV site resistance on the maximum IR-drop of two sets of our prototype layout. Note that TSV site resistance is on a log scale. The solid lines represent the core tiers and the dashed lines represent the memory tiers.

tiers as well as the top (furthest from the supply bumps) two memory tiers. In this graph, the TSV site resistance begins to affect the IR-drop scaling behavior of the clustered TSV topology. Again, the more correlated nature of the IR-drop between the various tiers in the distributed TSV topology is evident. The crossover point when the distributed topology produces higher IR-drop than the clustered topology has shifted to the left compared to Figure 87. This effect can also be seen in the context of Figures 78 and 80.

Further increasing the number of sets to the maximum examined, ten sets stacked together, we again repeat the resistance scaling simulations in Figure 89. To maintain readability, we only show the results for three core tiers, cores 1, 5, and 10. The same trends as in the previous graphs remain evident, though there are several interesting observations to be made. It is interesting to note the change in scale on the dependent axis (IR-drop) between Figures 87 and Figure 89. The maximum IR-drop plotted increases from 40 to 800mV. Also, the variation between the tiers in the clustered TSV topology become even

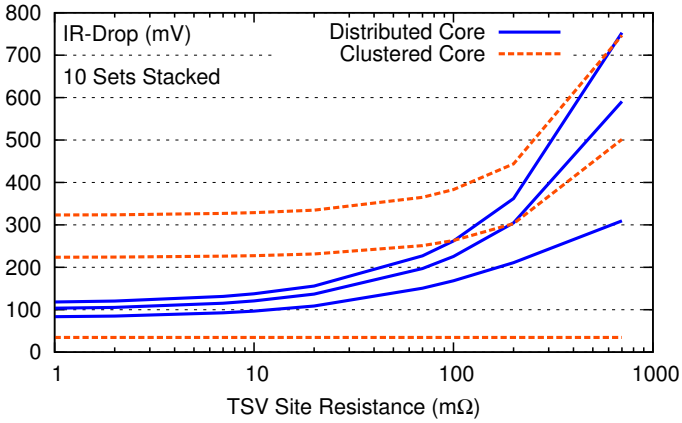


Figure 89. The effect of TSV site resistance on the maximum IR-drop of ten sets of our prototype layout. Note that TSV site resistance is on a log scale. The IR-drop of only cores 1, 5, and 10 are shown.

more extreme in the case with 30 tiers stacked together. The difference between the maximum and minimum is more than $700mV$ when the TSV site resistance is $700m\Omega$.

5.5.4 Possible Electro-Migration Issues

Electro-migration has become an increasingly important consideration in deep sub-micron IC design. Figure 90 shows the current density in the TSVs for the clustered (left) and

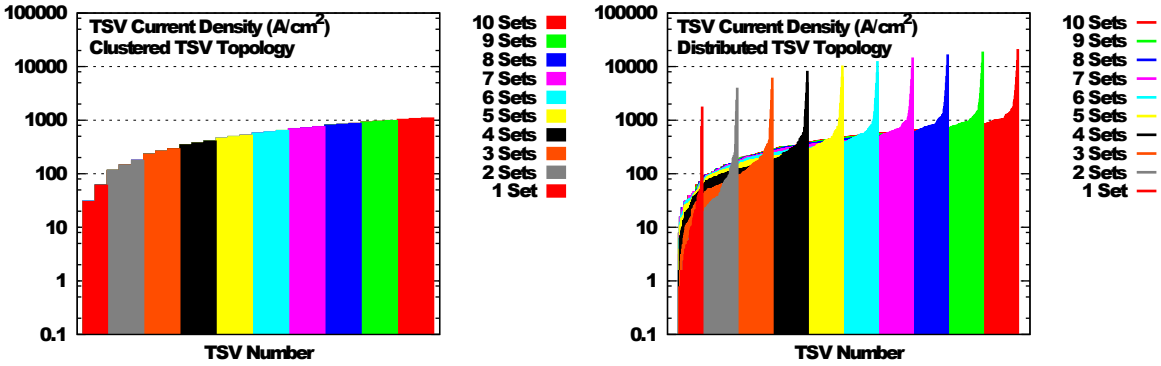


Figure 90. The TSV current density for the clustered topology. The current density numbers are sorted in increasing order from left to right.

distributed (right) TSV topologies. The baseline TSV dimensions are assumed: $6\mu m$ square diameter and $20\mu m$ length. The maximum current density in the TSVs configured in the clustered topology is much lower than for the TSVs in the distributed topology. This can be mitigated by increasing the TSV count in the TSV sites over the C4 bumps, which have the highest current density. It should be noted that for this particular case the current density is

far below the limits of most commonly-used TSV conductors.

5.5.5 Decreasing C4 Bump Pitch

The Euclidean distance between neighboring C4 bumps in our default layout is 400μm. Given the low power dissipation of a single core this is sufficient for low-tier systems, however, for our 1000-core system, the IR-drop is still above the 10% noise margin, even using the distributed TSV topology. In this and the following subsections we examine several methods to reduce the IR-drop and dynamic noise for our 1000-core system to meet the requirements. Figure 91 shows the IR-drop when the C4 bump pitch is reduced

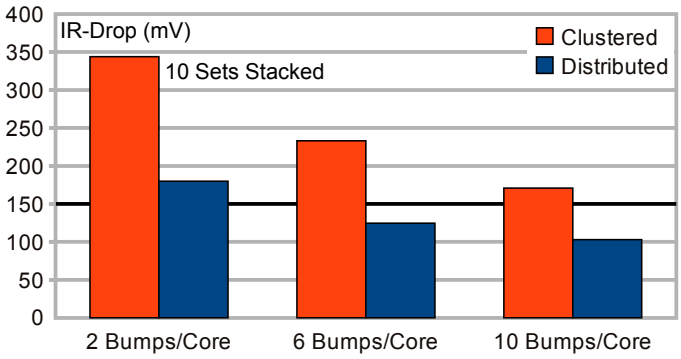


Figure 91. The maximum IR-drop for ten sets of our prototype layout stacked together with increasing numbers of C4 bumps per core added.

to allow 6 and 10 bumps per core. As the figure shows, it is possible to reduce the IR-drop for our 1000-core system below the 10% noise margin, 150mV, by adding 6 or more C4 bumps per core, which translates to a C4 bump pitch below 200μm. Also of note, lower IR-drop is achieved by using the distributed TSV topology than by halving the C4 bump pitch with the clustered TSV topology. Figure 92 shows the same comparison for dynamic noise. The trends are very similar to the trends for IR-drop.

5.5.6 Adding Decap Tiers

We have demonstrated that supply noise scaling in many-tier systems can quickly become a limiting factor in their design. Even for our relatively low-power 1000-core processor the supply noise is still above the 10% noise margin for the baseline system. In this subsection

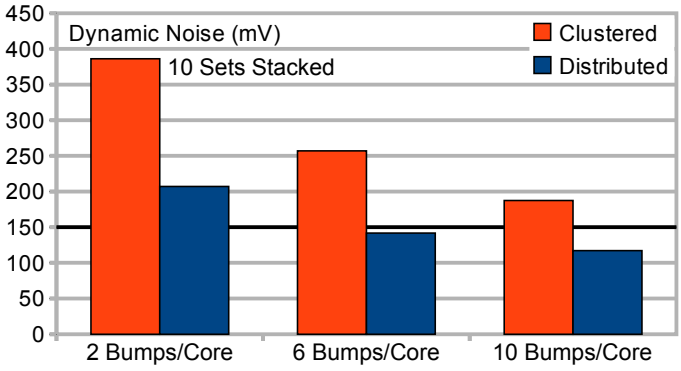


Figure 92. The maximum dynamic noise for ten sets of our prototype layout stacked together with increasing numbers of C4 bumps per core added.

we analyze the addition of layers containing only decoupling capacitance. We created the layout for this tier using the same 130-nm process used in the rest of our analysis, however, we expect that real systems incorporating decap tiers will use processes that enhance the capacitance per unit area while reducing production cost. Our decap tier contains a total of nearly $0.18nF$ per core, for a total of $18nF$ per tier, at a density of $0.57 fF/\mu m^2$. Figure 93 shows the improvement generated by adding either a single decap tier *per system* or one

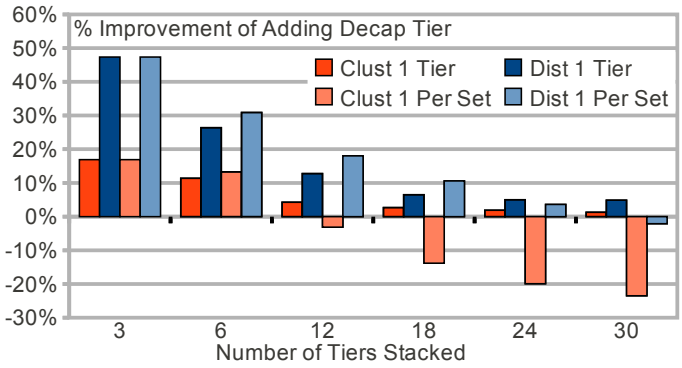


Figure 93. The improvement in dynamic noise created by adding either a single decap tier per system (1 Tier) or one decap tier per set (1 Per Set) to our scalable processor.

decap tier *per set* over the same system without decap tiers. In this case the decap tier is always added to the set on the side that is farthest from the power-supply bumps. The figure shows that the distributed TSV topology always benefits more from extra decap than the clustered topology. It also shows that beyond four sets stacked (12 tiers) the clustered topology results in *worse* power-supply performance when adding one decap tier

per set. The extra decoupling capacitance of the decap tiers cannot overcome the additional TSV parasitics that are introduced by adding decap tiers to the system. Also, adding one decap tier per set for systems with eight (24 tiers) or more sets stacked, results in lower improvement than adding a single tier per system for the distributed TSV topology.

Figure 94 shows the improvement over the default case of adding increasing numbers

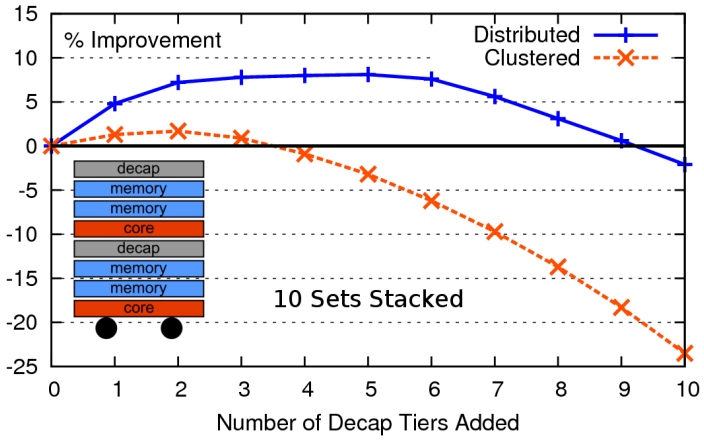


Figure 94. The improvement in dynamic noise created by adding increasing numbers of decap tiers to the 10 sets stacked system.

of decap tiers distributed throughout the 3D stack for the system with ten sets stacked. The results show that adding increasing numbers of decap tiers very quickly degrades the performance with the clustered TSV topology. However, for the distributed TSV topology the performance increases for a time before finally succumbing to the increased TSV parasitics of the decap tiers.

5.5.7 Pass-Through TSVs

Finally, we examine TSVs that pass-through the lower tiers without connecting to their power grids. These TSVs are meant to supply power only to the higher tiers in the stack. This trades off some additional lateral IR-drop in the lower tiers for lower maximum IR-drop in the system as a whole. A physical depiction of this design approach is shown in Figure 95 for the two TSV topologies. Figure 96 shows the impact on maximum IR-drop of increasing stacking. Figure 97 shows the maximum dynamic noise with increased

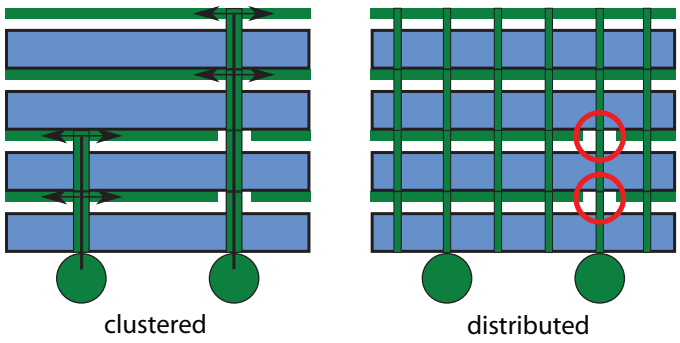


Figure 95. A side view of a 3D stack with pass-through power distribution TSVs. The TSVs connected to the C4 bump on the right do not connect to the distribution wiring on the lower two tiers.

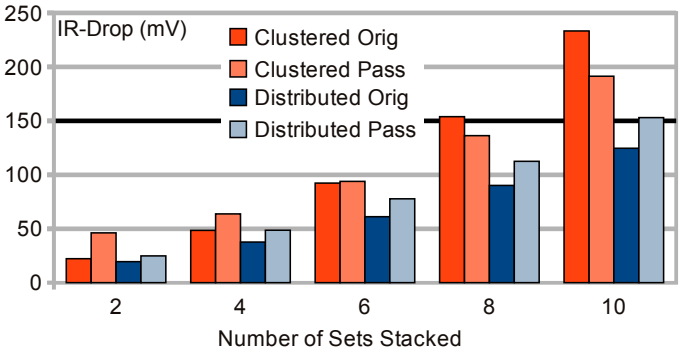


Figure 96. The maximum IR-drop with pass-through power distribution TSVs. The results are for a case with 6 bumps/core.

stacking. The simulations are for a case with 6 bumps/core. The results show that this technique is beneficial for large stacks with the clustered TSV topology, and reduces IR-drop for the 10 sets stacked case by nearly 18% and dynamic noise by nearly 17%. This style of implementing pass-through TSVs does not improve noise for systems with the distributed TSV topology.

There are many possible connection topologies for pass-though TSVs in combination with the distributed TSV topology. After some searching we found a set of connections that result in lower supply noise for the distributed TSV case. Figure 98 depicts this topology. In the distributed topology, the memory layers provide noise slack and decap to the core layers. Passing through some of the lower-level core layers therefore allows more of the memory layers to lower the noise level of the highest core layer, which has the maximum noise in the system. There remain several parameters, such as number of TSVs that

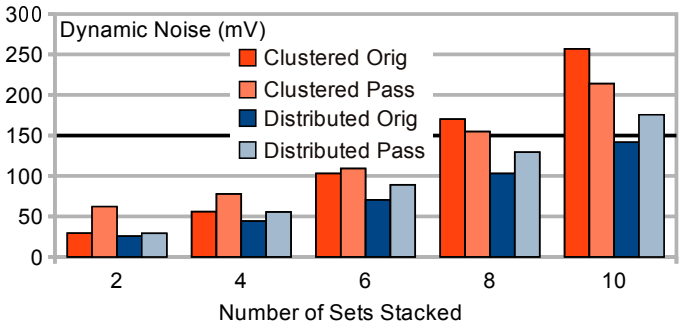


Figure 97. The maximum dynamic noise with pass-through power distribution TSVs. The results are for a case with 6 bumps/core.

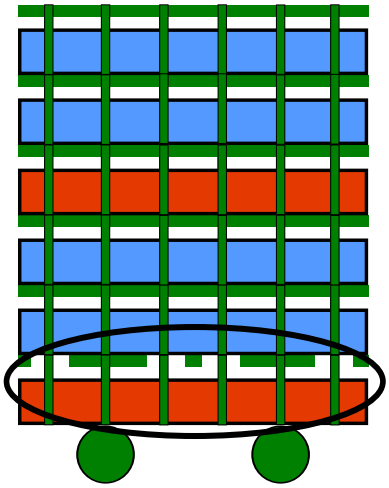


Figure 98. A side view of a 3D stack with an alternative connection topology of pass-through power distribution TSVs for the distributed TSV topology. The TSVs not connected to the C4 bumps do not connect to the distribution wiring on the lower core tier (orange).

pass through and the depth of the stack that they pass through, that are related to this pass-through TSV connection topology and that effect the final noise performance of the system. The affect of these parameters is relatively small, but we demonstrate the use of this design style to show that pass-through TSVs can still be beneficial for the distributed TSV topology. Figure 99 shows the dynamic noise results of using various number of TSVs passing through various numbers of core layers below the uppermost core layer. The figure shows that passing through more core layers always results in decreasing dynamic noise. Additionally, there is an optimal number of pass-through TSVs that should be used. If too few are used the benefits are very small, and if too many are used the lower core tiers begin to exhibit higher maximum noise.

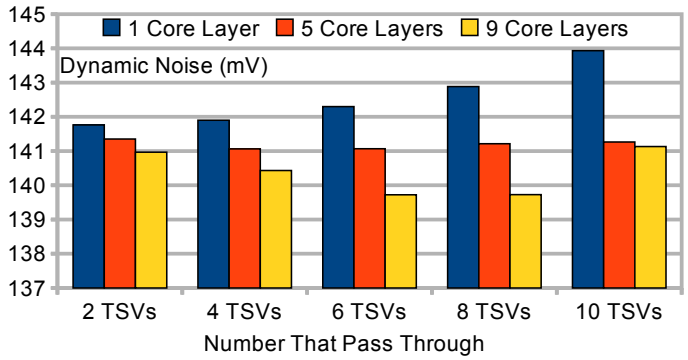


Figure 99. The maximum dynamic noise with alternative pass-through power distribution TSVs for the distributed TSV topology. The number of core layers that are passed through varies, as well as the number of non-C4 TSVs that pass-through. The results are for the 10-sets stacked case with 6 bumps per core.

5.6 Summary

This chapter explores 3D power-delivery-network design and shows that both IR-drop and dynamic noise can be improved in these systems by exploiting the particular attributes of power-supply TSVs that are unique compared to those of C4 supply bumps. Previous works have assumed a straightforward extension of traditional power-supply-network design in which the TSVs are treated as an extension of the C4 bumps. We advocate a design style in which power network TSVs are distributed with small pitch (relative to the package bumps) throughout the entire surface of the layout. This design style increases the level of coupling between the power distribution networks of the various tiers in the 3D stack. Thus allowing the utilization of decoupling capacitance, and IR-drop and dynamic noise slack in the lower-power tiers to reduce maximum system-level IR-drop and dynamic noise.

To support our claims we designed a 1000-core 3D processor across 30 stacked tiers at the layout level. Our 3D IR-drop analysis method was verified against commercial-grade sign-off IR-drop analysis software from a major EDA vendor at both the 2D and two-tier 3D level. Detailed simulations of the stacking scaling and TSV resistance scaling demonstrate that the distributed TSV topology generally provides much lower IR-drop and dynamic noise. In our baseline system with 30 stacked tiers, the distributed topology provides nearly 50% lower IR-drop and 42% lower dynamic noise than the clustered topology. For low-tier

systems the savings are still significant. In fact, the distributed TSV topology lowers IR-drop for a 3-tier system compared to a non-3D system by 21%, and dynamic noise by 32%, even though the total power consumption is higher in the 3-tier system. We also examine several techniques to reduce power-supply noise and their effects on both the clustered and distributed TSV topologies.

CHAPTER 6

DESIGN AND ANALYSIS OF 3D-MAPS

The potential of 3D IC stacking has been examined by researchers for many years. Only recently has the increasing cost of continuing process technology shrinks, and the incredible memory-bandwidth demand of multi- and many-core systems brought 3D technology to the forefront of commercial interest. Several large manufacturers as well as many smaller companies are actively investing in and investigating 3D stacking technologies and their benefits and challenges [117, 118, 119]. One major challenge comes from the currently limited selection of industrial-grade EDA tools that support the design and analysis of 3D systems.

In this chapter we demonstrate our methodology for designing and analyzing 3D-MAPS (3D MAssively Parallel processor with Stacked memory), a 64-core 3D-stacked memory-on-processor system. For every step of the design process we address the specific issues that 3D designers will encounter when dealing with tools that are not specifically designed to meet their needs. There are several works presented in the literature that describe various 3D architecture design options and physical design algorithms for 3D ICs, but very few in the area of 3D design demonstration and methodology. Thorolfsson *et al.* [73] described the design of an FFT processor with 3D stacked memory implemented with MIT Lincoln Lab's 3-layer process [75]. However, they do not discuss cross-talk or power-noise analysis in 3D systems and do not include a thermal analysis. The contributions of this chapter are as follows:

- The design of 3D-MAPS, arguably the first many-core 3D processor in academia. Our 3D processor contains 64 5-stage pipelined, 2-way in-order VLIW cores that were fully custom designed. Two dies are stacked in 3D-MAPS, one 64-core die and one SRAM die. Each core owns a dedicated 4KB SRAM tile, which is stacked above the core and connected using face-to-face 3D vias. Our architecture is verified with

several multi-core benchmarks. 3D-MAPS demonstrates memory bandwidth up to 63 GB/s based on our many-core benchmarks.

- The detailed methodology used to construct the physical layouts of the 3D-MAPS processor and how to perform various 3D analysis. Our tool-flow is based on commercial tools from Cadence, Mentor Graphics, and Synopsys, and is enhanced with various add-ons we developed to handle TSVs and 3D stacking. We provide sign-off 3D timing, power, thermal, IR-drop, signal integrity, and clock waveform analysis results based on DRC/LVS-passed 3D-GDSII layouts.

3D-MAPS was taped-out in March of 2010 and will be fabricated using 130nm Chartered Technology and Tezzaron TSV/3D Technology. Once the fabrication and package/board design are completed in September 2010, our simulation results will be verified using measured data.

This chapter describes the design and analysis of 3D-MAPS, a collaborative effort of 19 contributors [120]. The contributions made by this dissertation are detailed in sections 6.3.1, 6.4.2, and 6.4.4.

6.1 3D Stacking Technology

The 3D-MAPS processor will be fabricated using a six-metal 130*nm* process provided by Chartered Semiconductor that is specially modified to include through-silicon vias (TSVs) according to the specification of Tezzaron Semiconductor. The TSVs are manufactured in a front-end-of-line process, better known as via-first. Trenches are dug into the silicon and filled with tungsten. Then devices and metal layers are patterned. Next, one wafer is flipped over and thermo-compression bonded to another wafer. Finally, one wafer is thinned until the trenched TSVs are revealed from the backside. This produces a two-layer face-to-face bonded stack that uses TSVs for IO. Because the wafers are bonded before thinning, there is never a need to handle a thinned wafer. Figure 100 shows a diagram of the completed die stack.

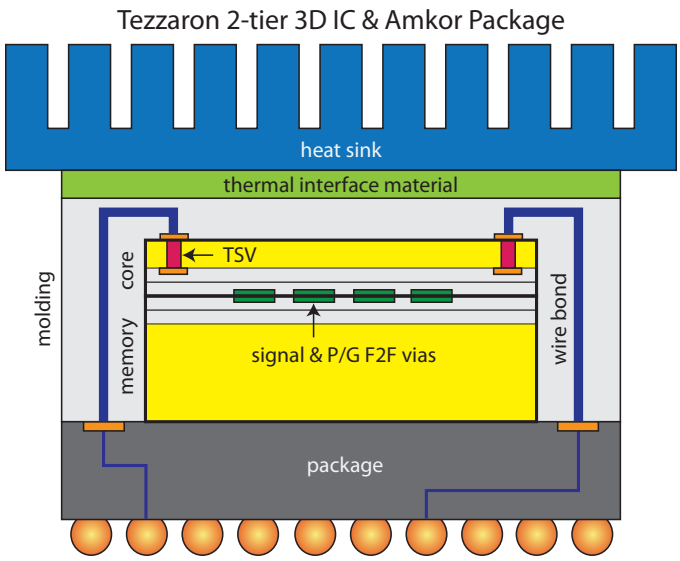


Figure 100. Side view of the final stacked dies based on Tezzaron's F2F and TSV stacking technology

The Tezzaron process produces TSVs that are approximately $1.2\mu m$ wide with $2.5\mu m$ minimum pitch and $6\mu m$ height. The face-to-face (F2F) connection, which is used for the main die-to-die communication, uses $3.4\mu m$ Metal 6 pads with $5\mu m$ pitch. The TSVs have a parasitic resistance of around $600m\Omega$ and a parasitic capacitance of about $15fF$. The F2F connection has negligible resistance and capacitance, about the same as a local via. The 3D-MAPS die footprint is $5 \times 5mm$.[1] Therefore, there are one million total face-to-face connections. TSVs are used solely for IO purposes in the current design. The 3D-MAPS design uses $72,576$ F2F connections for power and signals, and approximately $1,800$ TSVs for IO. The 130nm Chartered standard cell library provided to us includes only peripheral-style IO. For that reason, we include functional TSVs only underneath the IO-cell pads. Dummy TSVs are inserted in the middle of the die to ensure that planarity requirements are met. More details on our F2F vias and TSVs are presented in Section 6.5.2.

6.2 3D-MAPS Architecture

The 3D-MAPS processor, illustrated in Figure 101, contains 64 cores laid out in an 8×8

[1]This is the space assigned to us as part of the 2009 DARPA/Tezzaron multi-project wafer run, and this restricts the number of cores and memory capacity.

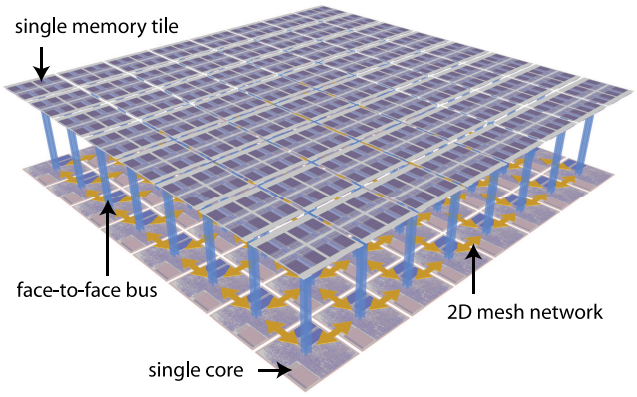


Figure 101. Two-die 3D-MAPS processor with 64 cores and 3D stacked memory

grid and connected with a 2D mesh communication fabric. Each core is shown in the figure connected by F2F connections with its local memory store. Each core has a $570 \times 570\mu m$ footprint and the entire 64-core die has a $5 \times 5mm$ footprint area. The entire processor operates synchronously at $277MHz$. Details of the 3D-MAPS architecture are available in Healy *et al.* [120].

6.3 Physical Design Methodology

Figure 102 shows the overall physical design flow used to produce the single core and sin-

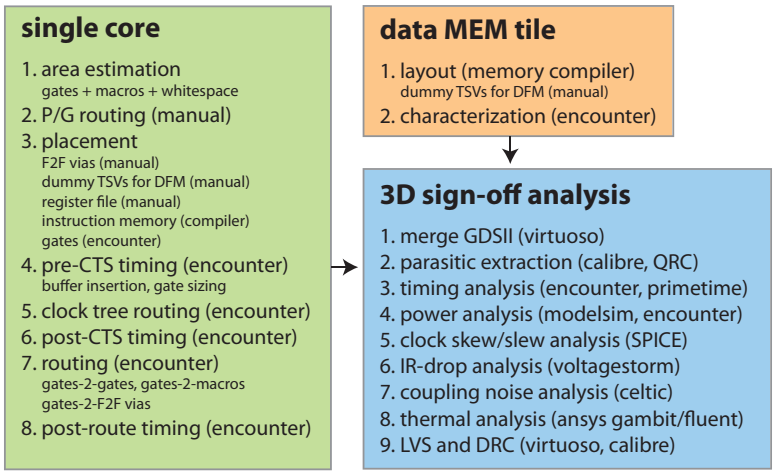
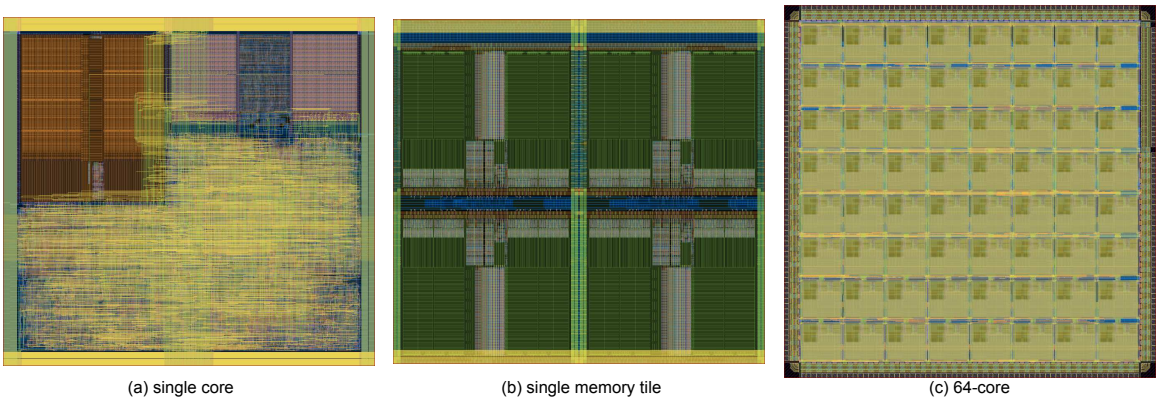


Figure 102. Our flow for the design and analysis of single core and single memory tile stack.

gle memory tile layouts for the 3D-MAPS processor. The physical design flow begins with an RTL description of the processor core written in VHDL. We then use Synopsys Design

Compiler to compile the VHDL into structural Verilog for each die. The compiled Verilog is then input to Cadence Encounter to perform the automated physical design steps. We use Cadence Encounter to perform gate placement, sizing and buffering optimization, signal routing, clock routing, and power and ground network generation. We also use many of the tools integrated into Encounter to perform early analysis on the design to ensure reasonableness before sign-off analysis is undertaken. However, Cadence Encounter and its integrated point-tools do not understand F2F vias, TSVs, and 3D stacking, i.e., multiple die definitions. Thus, we have developed several add-ons and found ways to manipulate LEF/DEF and other intermediate files to manage F2F vias, TSVs, and 3D stacking. More details are provided in the subsequent sections. The memory bank, tile, and die designs are done with a memory compiler provided by Artisan. The remaining part of this section describes each step of the physical design in detail for 3D-specific considerations. Section 6.4 provides details on 3D sign-off analysis.

The many-core-level physical design occurs in much the same way. The array of 64 cores takes up almost the entire core layer, all the remaining space is allocated to IO cells and power and ground network distribution. The space between the cores is used to place the test circuitry. Figure 103(a-c) shows the layouts of single core, single memory tile, and 64-core dies.



(a) single core (b) single memory tile (c) 64-core

Figure 103. Various layout views of the 3D-MAPS processor.

6.3.1 3D Power and Ground Network Generation

The power and ground distribution networks are generated mainly using the stripe and ring generation commands in Cadence Encounter. At the single-core level, the instruction memory and register file have individual power and ground rings. The core also has a set of rings. The module and core rings are connected near the corners. At the many-core level, each core is connected by continuing the stripes that make up the core rings. There is also a top-level set of rings that connects the IO supply pads and the single-core stripes.

In the memory layer, each bank is generated with its own set of power and ground rings. These rings are then connected using a single-core-sized ring and stripes where needed. The goal is to have the rings on both the core layer and memory layer line up. By lining up these rings we can connect them in places with available upper-level routing space using the vast array of face-to-face (F2F) connections. At the top-level of the hierarchy in the memory layer all the individual memory-tile rings are connected similarly to the core-layer. Again, the goal is to have as much of the power and ground top-level distribution wiring line up as possible. This allows the creation of a very low resistance connection for power and ground distribution to the memory layer.

Decoupling capacitors (decaps) are inserted into the design using Cadence Encounter. This is done after routing and optimization to ensure the most optimizing timing possible. In the memory layer, we insert a large amount of decaps on the power rings in the blank space around the memory banks. This allows the memory layer to provide large amounts of on-demand current to the cores. Decaps were also inserted at the multi-core level in between the cores and testing circuitry. The core layer contains 109,236 individual decaps with a total of $1.09nF$ of capacitance. The memory layer contains 82,048 individual decaps with a total of $0.82nF$ of capacitance.

6.3.2 F2F Via and TSV Placement

Communication between the core and memory dies occurs through the face-to-face (F2F) vias. Any net that connects to a F2F via, and thus circuitry on the other die, is called a

3D net. The individual design for each wafer therefore must contain pins for all nets that cross the F2F boundary. The memory layer contains only the data memory banks and their connections. Accordingly, we first fix the location of the memory banks, then we manually place pins in both dies directly above the pins on the memory banks. This causes some replication of pins, mainly the clock, write-enable, and address pins. The F2F vias are mainly used to connect the core and memory logic, therefore, most F2F vias are placed at the single-core level. The TSVs are used only for off-chip IO in the current version of 3D-MAPS. The foundry-supplied IO cell library is peripheral-style only. Therefore, the only electrically active TSVs are placed inside the IO cells underneath the bond-pad.

The Tezzaron TSV process imposes a unique requirement: a mandatory maximum TSV pitch of $250\mu m$ throughout the entire layout region. This means that there needs to be at least one TSV inside every $250 \times 250\mu m$ window. This requirement is in place to maintain the planarity of the wafer during chemical and mechanical polishing (CMP). Because the 3D-MAPS cores are $560 \times 560\mu m$ and we do not use TSVs inside the core region, we must place a 3×3 array of dummy TSVs inside each core to meet this maximum pitch requirement. The dummy TSVs occupy an area of about 18 minimum-sized inverters inside each core. Since Tezzaron's via-first TSVs occupy the device layer as well as the bulk, dummy TSVs should not interfere with gates. Thus, our strategy is to place dummy TSVs manually before gate placement at the single-core level. The large size of the instruction memory module implies that dummy TSVs need to be inserted inside the module. We found a small white space in the module that can accommodate a single TSV. Section 6.5.2 provides more details of the layouts.

6.3.3 3D Placement and Routing

Cadence Encounter is used to perform placement at both the many-core level and the single-core level. The 3D connection information is propagated to the placer through the use of fixed pins on Metal 6 representing the F2F connections. These pins constrain the placement to optimize correctly for the full 3D system. At the single-core level, we fix the pins

at the edge of the core related to the NEWS core-to-core communication to ensure short wirelengths at the many-core level. At the many-core level, we fix the cores into a regular 8×8 grid and treat them as constraints for optimizing and placing the gates for the global signals and test circuitry.

Cadence Encounter is used to perform sizing and buffering optimizations, and NanoRoute is used to perform routing. The 3D connection information is propagated to the optimizer through the use of back-annotation of capacitance and arrival time requirements on the fixed pins. These constraints force the optimization engine and the router to account correctly for both sides of the 3D nets.

6.3.4 3D Clock Routing

We perform clock routing using the clock tree synthesis functions of Cadence Encounter. The clock network is contained mainly within the core layer. Each memory bank in the memory layer has a clock pin that is propagated to the core layer using a fixed F2F connection. This pin is annotated with the capacitance of the routing inside the memory layer, as well as the gate capacitance of the clock pin on the memory bank itself. This minimizes the clock skew between the single core and memory tile stack. At the many-core level, each core has a single input clock pin that feeds its internal clock tree. The clock-to-sink delay inside each core is annotated onto this clock pin to ensure that the arrival times for the flip-flops inside and outside the cores are well matched.

6.4 3D Sign-Off Analysis

The existing Cadence, Synopsys, and Mentor Graphics tools are designed for 2D ICs and do not handle 3D designs and TSVs. The following sections describe our strategy to extend these tools to analyze and verify 3D-MAPS.

6.4.1 3D Timing and Signal Integrity Analysis

Our 3D timing analysis is based on Synopsys PrimeTime [73]. First, we prepare the Verilog netlist files of both dies and the SPEF files containing extracted parasitic values for all the nets of the dies. Then, we create a top-level Verilog netlist that instantiates each die's design and connects the 3D nets using F2F connections. We also create an SPEF file that has a parasitic model of the F2F connections. After that, we run PrimeTime with all the Verilog files and the SPEF files to get the timing analysis results. The timing on 3D nets can be easily checked by specifying the *'-through'* argument with the F2F names to the *'report_timing'* command. We use the obtained 3D timing values to optimize the 3D nets.

3D signal integrity analysis must also contain a 3D component because nets may have enough coupling capacitance to be considered a problem only when all dies are considered simultaneously. For signal integrity analysis, we use Cadence CeltIC. Again, we input an SPEF file that contains the information for both dies and the parasitics from the F2F connections. Then with the merged Verilog netlist, CeltIC finds all the paths with noise violations, including the 3D nets.

6.4.2 3D Power-Noise Analysis

We perform 3D power-noise analysis using Cadence VoltageStorm. The stand-alone VoltageStorm takes in a DEF file, technology files, and power dissipation files to generate both peak and average power noise values. Performing this analysis for a 3D design is a challenge, because VoltageStorm does not understand multi-layer designs. One method of analysis is to assume a small number of connections between each layer of the die stack for the power and ground nets near the periphery of the chip (near the IO cells). Then VoltageStorm analysis is performed on each tier individually. In this case the power noise threshold should be lowered to ensure that the devices are always provided with a sufficient voltage level.

For our design, we perform true 3D power-noise analysis with VoltageStorm. To accomplish this we compile a technology file that contains all of the 3D layers. This technology file contains multiple copies of each metal layer, one for each layer in the 3D stack. Then, 3D DEF files are constructed from the design of each layer. A separate LEF file must also be constructed that contains instances specific to each layer. Finally, this information is fed into VoltageStorm to obtain true 3D power-noise values. For 3D-MAPS, we created a technology file with duplicate metal layers for all but Metal 6, which is used for the F2F connections. This is appropriate because there is no interposer between the adjacent Metal 6 layers and they have very low contact resistance after the thermo-compression bonding step. Section 5.3 provides a more detailed explanation of the 3D power-noise-analysis methodology.

6.4.3 3D Thermal Analysis

3D designs have the potential to suffer from significant thermal problems due to the higher thermal resistance between certain active silicon layers, the heatsink, and heatspreader. We use ANSYS' Gambit and Fluent packages for our thermal analysis. Gambit is a meshing and model generation software that sets up thermal analysis problems. Fluent is the simulation engine that calculates the thermal distribution of the chip using the model generated by Gambit.

We first simulate a single core to get an accurate view of the thermal map at a detailed level. Gambit is used to model the 3D chip-stack and includes both core and memory layers, as well as a model of the rest of the package and a 5mm tall heatsink. The stack is first divided into numerous thermal layers like gates, poly, Metals 1-6, via, dielectric, *etc*. Each of the thermal layers are meshed into 256×256 mesh grid points. To determine the material properties for each mesh volume, the GDSII file is parsed to determine the correct ratio between the various materials of each layer at each particular grid point. For instance, the parsing of Metal 3 mesh volumes would return the ratio of metal to dielectric for each grid point by determining where the Metal 3 rectangles exist in that grid volume. This

ratio is used to calculate the weighted average of the material properties and to determine the effective thermal conductivity of that grid point. Power sources, determined by the power consumption map are then inserted into each mesh volume. Finally, Fluent is used to perform a steady state thermal analysis to calculate the thermal map.

This procedure is then repeated at the many-core level to provide a temperature profile of the overall chip-level thermal map. At the many-core level, the model also includes C4 bumps at the bottom of the stack, as well as I/O cells around the core area.

6.4.4 3D DRC and LVS Verification

3D design rule checking (DRC) is generally applied to GDSII layers that connect two dies or connect a die to the packaging substrate. However, the F2F connection in the Tezzaron process is created simply by including Metal 6 in the same location on both dies. In this case, only GDSII layers that interact with the TSVs are necessary to pass DRC. For 3D-MAPS we use a DRC ruleset from Tezzaron that contains DRC rules for the TSVs. The TSVs are also included as a layer in the GDSII files. We use Calibre from Mentor Graphics to perform DRC for 3D-MAPS.

3D layout versus schematic (LVS) is performed by running Calibre with a combined GDSII file and a 3D-aware ruleset file. For the layout data, we combine the two GDSII files into one, and for the memory layer, the original GDSII numbers are mapped into a non-overlapping space. For the schematic data, we prepare a netlist by combining the two Verilog netlists for both dies. Then, Calibre uses the 3D-aware ruleset to check the match between the layout and schematic, including the 3D connections.

6.5 Layout and Analysis Results

6.5.1 Architectural Simulation Results

Table 11 shows the results from our many-core architectural simulations of the 3D-MAPS

Table 11. Architectural performance metrics for 3D-MAPS.

Benchmark	Memory Bandwidth (GB/s)	IPC per core	BIPS
string_search	8.9	0.65	11.52
matrix_multiply	13.8	0.32	5.67
median	63.8	1.62	28.72
aes_encrypt	49.5	0.97	17.20
motion estimation	24.1	1.20	21.27
histogram	30.3	0.90	15.96
edge detection	15.6	0.95	16.84
k-means	40.6	0.94	16.66

processor. The table reports memory bandwidth in gigabytes per second (GB/s), instructions per cycle (IPC), and billions of instructions per second (BIPS). The 3D-MAPS processor achieves memory bandwidth up to 63.8 GB/s, which is higher than that of a modern Intel Core i7 processor and comparable to the memory bandwidth of a high-end GPGPU running at four times the frequency with much larger area.

6.5.2 Physical Layouts

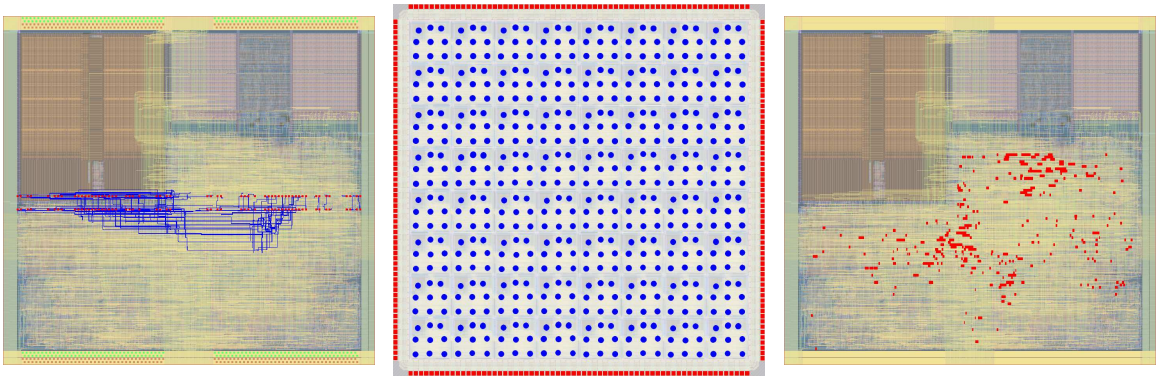
Table 12 shows a summary of the characteristics of the 3D-MAPS layout. Figure 103

Table 12. Physical design summary.

Process technology	Global Foundries 130nm
Die size	$5 \times 5mm$
Core footprint	$560 \times 560\mu m$
Core-to-core pitch	$570\mu m$
PG 3D connections/core	668
Total PG 3D connections	$42,752$
Data 3D connections/core	116
Total data 3D connections	$7,424$
TSVs/IO pad	204
Total IO TSVs	$47,940$
Dummy TSVs	$6,540$
Total maximum IR-drop	$78mV$
Maximum operating frequency	$277MHz$

shows various layout views of the 3D-MAPS processor. The core footprint is $560 \times 560\mu m$. The layout of one tile of SRAM memory is also shown. A single tile contains 4 banks of 1KB data memory. Thus, the total SRAM data memory capacity of 3D-MAPS processor is $4KB \times 64 = 256KB$. The full many-core layout of the core layer has dimension of $5 \times 5mm$. Each core is arrayed in an 8×8 grid and core-to-core communication occurs using short

wires. The core-to-core pitch is $570\mu m$.



(a) face-to-face vias and wires (b) IO and dummy TSVs (c) buffers added for timing

Figure 104. Layout views of the 3D-MAPS processor highlighting various areas of interest.

Figure 104(a) shows the F2F connections used for the 3D communication (red) and power and ground network distribution (orange and green). The nets connected to the F2F communication pads are highlighted in blue. There are $1,018$ power and ground F2F connections per core, for a total of $65,152$ power and ground F2F connections over the entire die. Each core also uses 116 F2F connections for signals and clock, for a total of $7,424$ F2F connections over the entire die. Figure 104(b) denotes the location of the TSVs in the core layer. The dummy TSVs, shown in navy, are located inside the cores.[2] The IO TSVs shown in red are located around the periphery. There are $1,784$ TSVs used for IO and 576 dummy TSVs. Timing optimization inserted 970 buffers into each of the cores. The buffer distribution is shown in Figure 104(c).

6.5.3 3D Sign-Off Analysis Results

The IR drop inside a single core is shown in Figure 105(a). The total drop is about $13mV$ inside one core. There are thick $10\mu m$ lines used for the core ring. These rings are connected at the top level using long stripes that run from edge to edge in both the x and y directions. The IR drop inside a single memory tile is shown in Figure 105(b). The total drop is about $10mV$. The 64-core maximum IR-drop is $78mV$. These values include true 3D-aware IR-drop analysis using sign-off-level Cadence VoltageStorm software.

[2]These dummy TSVs are enlarged on purpose for visibility.

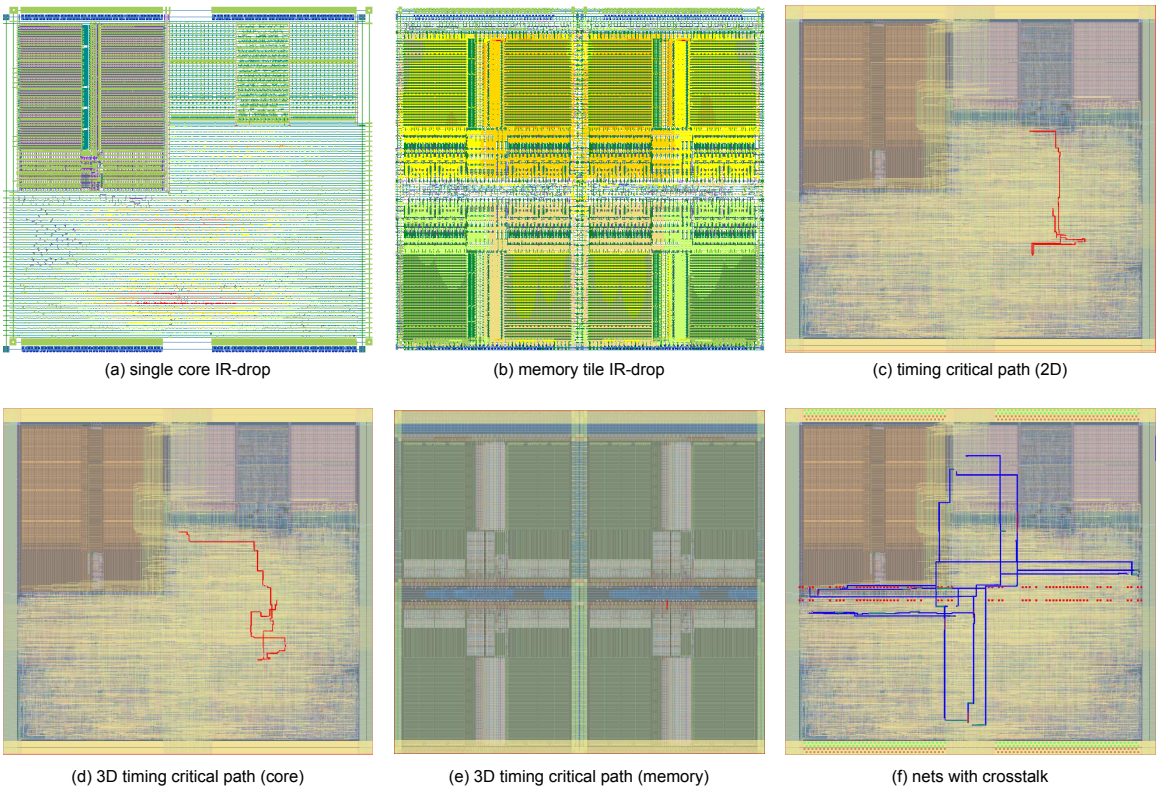
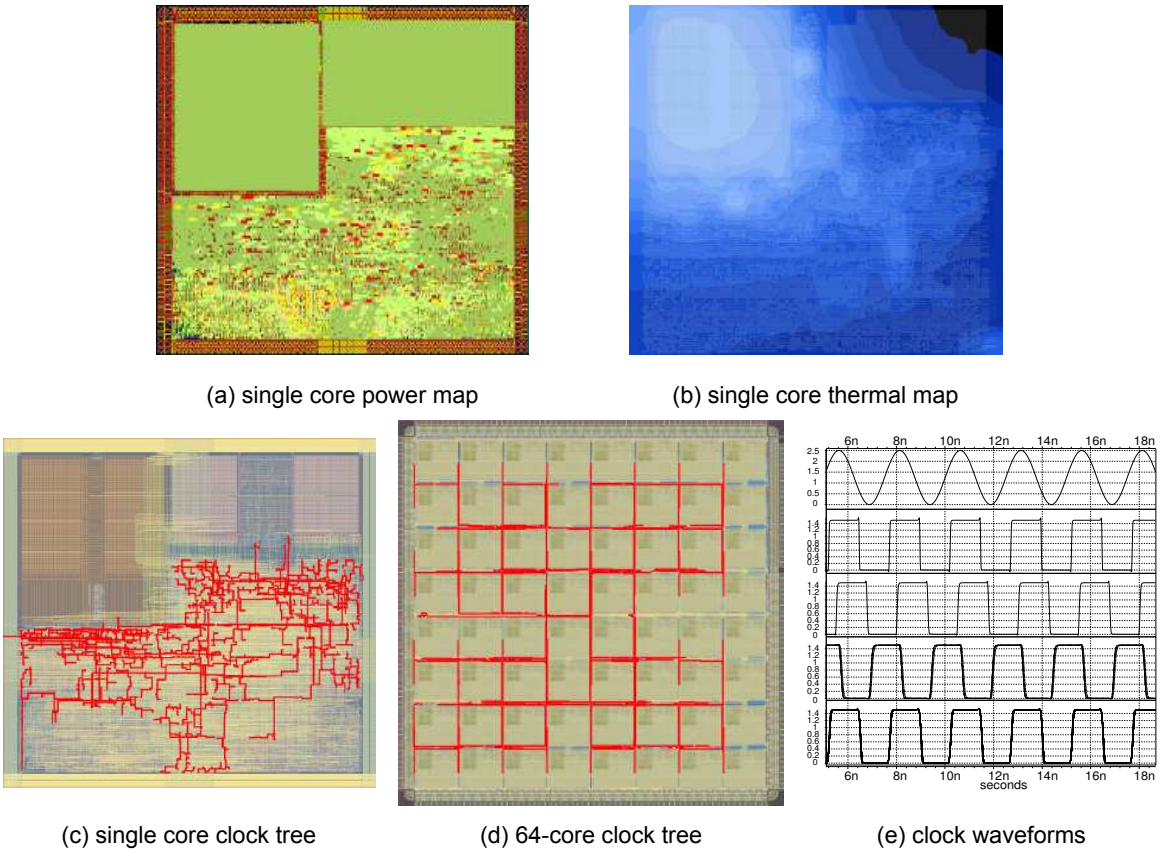


Figure 105. Various 3D sign-off analysis results for the 3D-MAPS processor.

3D timing analysis reports that the 3D-MAPS processor is capable of running at $277MHz$. The timing critical path inside each core is shown in Figure 105(c). The timing critical path runs through several muxes and the multiplier. The longest delay for a 3D net is for the address bus, which has a sink in each of the four memory banks and thus has larger wire-length. The 3D timing critical path is shown in Figure 105(d) and (e). The net is driven through several muxes and the memory address generator before ending at the data memory address pins. In the memory layer, each net travels a very short distance before reaching the memory bank address pins. Figure 105(f) shows the top five 3D nets that experience the most switching noise inside each core. The maximum noise value on the worst net is $674mV$, which is very close to the noise limit. The next highest noise value is much lower at $592mV$. There are a few 2D nets that experience more noise than the noisiest 3D nets.

The power map for a single core simulated using the string-search benchmark is shown in Figure 106(a). The figure shows that the instruction memory consumes the largest



(a) single core power map (b) single core thermal map

(c) single core clock tree (d) 64-core clock tree (e) clock waveforms

Figure 106. Power, thermal, and clock analysis results for the 3D-MAPS processor.

amount of power. The single core thermal map using the string-search benchmark is shown in Figure 106(b). The temperature is highest near the west side of the core, which covers the instruction memory. The maximum temperature is approximately $47°C$.

Figure 106(c) and (d) show the clock distribution nets at the single- and 64-core levels, respectively. The clock signal skew at the multi-core level (between two core's clock input pins) is $45.3ps$. The skew between the flip-flops inside a single core is $22.7ps$. Therefore, the total flip-flop to flip-flop skew over the entire design is $68.0ps$. The slew for the clock signals is around $117ps$ for both rise and fall.

Figure 106(e) shows HSPICE simulations for the clock tree. The first waveform shows the sine wave input from the off-chip PLL at the IO voltage. The second waveform shows the output of the input buffer cell that converts the IO voltage sine wave to the square core-voltage clock signal. The third waveform displays the output of the clock drivers that drive

the input buffer's square signal to the clock distribution network. The final two waveforms show the clock signal at the input to the cores and then the flip-flops inside the cores.

6.6 Summary

This chapter presents the design, layout, and analysis of 3D-MAPS, a 64-core memory-on-processor 3D-stacked system. It was built from the ground up to demonstrate extreme memory bandwidth using 3D connections. The layout and analysis was performed using commercial tools with several custom add-ons to enable full 3D awareness. 3D-MAPS simulates correctly at $277MHz$ and verified architectural simulations show that it achieves memory bandwidth above 63 GB/s on selected benchmarks.

CHAPTER 7

CONCLUSION

Reliability of microprocessors and application-specific integrated circuits has become of increasing importance in recent years. Device reliability and lifetime are affected by both operating temperatures and power-supply noise in modern designs. The thesis of this disseration is that the automated design tools and design techniques presented herein enable higher reliability while maintaining architectural performance. As evidence supporting this thesis, this disseration presents:

- A thermal-aware microarchitectural floorplanner for 2D and 3D ICs.
- A thermal-aware microarchitectural floorplanner designed to work with multi-core systems.
- A power-supply-noise limiting microarchitectural run-time controller and controller-aware floorplanner.
- A thermal and power-supply-noise scaling study of many-tier systems.
- A power-supply TSV topology that significantly lowers IR-drop and dynamic noise in 3D ICs.
- Layout-level design and analysis methods utilizing commercial tools that were used to create a 3D IC sent for fabrication.

Extensive simulation shows that the tools and techniques presented in this dissertation provide trade-offs between lower operating temperatures and performance, or lower power-supply noise with negligible performance impact.

To conclude, we include a short discussion of the limitations of the research presented in this dissertation, and possible future directions of exploration, to place the entire work in

a broader context. The work presented here has a broad focus on 3D integration technology, however, there are several issues with 3D integration that have yet to be resolved. For example, increased stacking of untested ICs results in exponential decreases in system-level yield [121]. This naturally leads to a requirement for testing of unpackaged dies or wafers, which is an area of active research. It also leads to many issues with testing of circuits that will only be complete *after* 3D stacking. Additionally, the proper micro and system-level architecture required to realize the optimal cost/performance point using 3D stacking technology is also an area of active research. Furthermore, this dissertation includes results relating to the use of microfluidic heatsinks in 3D ICs. Microfluidic channels are a promising technology, but their impact on system-level reliability and cost-effectiveness remains unquantified.

There are several interesting areas of possible future exploration related to the work presented by this dissertation. Thermal-aware microarchitectural floorplanning results in performance degradation compared to a non-thermal-aware floorplan. However, modern high-performance microprocessors typically include dynamic management schemes that throttle performance when the temperature of the processor passes a threshold. An analysis of thermal-aware floorplanning that includes the effects of dynamic thermal management may reveal that the thermal-aware floorplan results in higher real-world performance. Additionally, this dissertation explores power-supply noise in large-scale 3D systems mainly from an on-chip perspective. Major work needs to be done on the package and board level to determine and/or expand their power delivery capabilities to match the heat-removal capacity of microfluidic heatsinks. Finally, this dissertation has demonstrated the potential usefulness of the distributed TSV topology in a uniform fashion. The benefit of this technique is largely dependent on the power-supply networks of the layers that are being connected by those TSVs. This implies the potential for optimization of both wire sizes and TSV placement that considers inter-layer effects.

APPENDIX A

LP FORMULATION

The following variables are used for the LP-based floorplanning formulation:

- N: set of all modules in the netlist.
- E: set of all nets in the netlist.
- x_i, y_i: location of module i.
- w_i, h_i: half-width and half-height of module i.
- a_i, g_i: area and delay of module i.
- $w_m(i)$, $w_x(i)$: minimum/maximum width of module i.
- $\lambda_{i,j}$: normalized profile weight on wire (i,j).
- $z_{i,j}$: number of flip-flops on wire (i,j) after insertion.
- $X_{i,j} = |x_i - x_j|$ and $Y_{i,j} = |y_i - y_j|$.
- $T_{i,j}$: normalized product of the temperature of modules i and j.
- A: aspect ratio of the chip.
- X_x : maximum x_i, Y_x: maximum y_i.
- C: target cycle time.
- d_r: unit length delay of repeated interconnects.

The LP floorplanner determines the values for the following decision variables: x_i, y_i, w_i, h_i, and z_{ij}. The following are the variables used for bipartitioning:

- $B(u)$: set of all modules at iteration u.

- $M_j(u)$: set of all modules in partition j at iteration u.
- $S_{j,k}(u)$: set of modules assigned to subpartition k ($k \in \{1, 2\}$ for bipartitioning) in partition j at iteration u.
- $(\bar{x}_{jk}, \bar{y}_{jk})$: center of subpartition k contained in partition j.
- r_j,v_j,t_j,b_j: the right, left, top, and bottom boundaries of partition j.

The LP-based slicing floorplanning problem is formulated as follows:

Minimize:

$$\sum_{(i,j)\in E} (\alpha \cdot \lambda_{ij} \cdot z_{ij} + \beta \cdot (1 - T_{ij})(X_{ij} + Y_{ij}) + \gamma \cdot X_x) \tag{22}$$

Subject to:

$$z_{ij} \geq \frac{g_i + d_r(X_{ij} + Y_{ij})}{C},\ (i, j) \in E \tag{23}$$

$$X_{ij} \geq x_i - x_j \text{ and } X_{ij} \geq x_j - x_i,\ (i, j) \in E \tag{24}$$

$$Y_{ij} \geq y_i - y_j \text{ and } Y_{ij} \geq y_j - y_i,\ (i, j) \in E \tag{25}$$

$$z_{ij} \geq 0,\ (i, j) \in E \tag{26}$$

$$w_m(i) \leq w_i \leq w_x(i),\ i \in N \tag{27}$$

$$x_i,\ y_i \geq 0,\ i \in N \tag{28}$$

$$X_x \geq x_i \text{ and } A \cdot X_x \geq y_i,\ i \in N \tag{29}$$

Boundary Constraints:

$$x_i + w_i \leq r_j,\ i \in M_j(u), j \in B(u) \tag{30}$$

$$x_i - w_i \geq v_j,\ i \in M_j(u), j \in B(u) \tag{31}$$

$$y_i + m_i w_i + k_i \leq t_j,\ i \in M_j(u), j \in B(u) \tag{32}$$

$$y_i - m_i w_i - k_i \geq b_j,\ i \in M_j(u), j \in B(u) \tag{33}$$

Center of Gravity Constraints: for $k \in \{1, 2\}$, $j \in B(u)$

$$\sum_{i \in S_{jk}(u)} a_i x_i = \sum_{i \in S_{jk}(u)} a_i \times \bar{x}_{jk} \tag{34}$$

$$\sum_{i \in S_{jk}(u)} a_i y_i = \sum_{i \in S_{jk}(u)} a_i \times \bar{y}_{jk} \tag{35}$$

There are three terms in the objective function shown in Equation (22): profile-weighted wirelength ($\lambda_{ij} \cdot z_{ij}$), thermal-weighted wirelength ($(1 - T_{ij})(X_{ij} + Y_{ij})$), and footprint area ($X_x$), where λ_{ij} is the profiled activity factor of the wire between modules i and j. The minimization of the first term improves IPC while the minimization of the second term stretches the distance of two modules, thereby reducing thermal coupling. The function $(1 - T_{ij})(X_{ij} + Y_{ij})$ was chosen as the temperature-dependent portion of the cost function because it satisfies several properties: it is linear with respect to distance between module i and module j, it considers the temperatures of both module i and module j, and it grows smaller when considering hot blocks and larger when considering cool blocks. Because the cost function is being minimized in the LP and not maximized, it is necessary to only consider minimization of the distance between cool blocks and not maximization of the distance between hot blocks, as would be preferable. Since minimizing $X_x \cdot Y_x$ (floorplan area) is non-linear, X_x is minimized because constraint (29) enforces $A \cdot X_x$ to be greater than all y values. α, β, and γ are user-defined parameters for weighting the relative importance of the performance, thermal, and area objectives. In the case that $\alpha = 0$, the floorplanner optimizes temperature and area only. In case $\beta = 0$, the floorplanner optimizes the performance and area objectives only. Lastly, for conventional area/wirelength-driven floorplanning the following objective function is used:

$$\gamma \cdot X_x + \delta \cdot \sum_{(i,j) \in E} (X_{ij} + Y_{ij}) \tag{36}$$

The area objective has a positive impact on performance and wirelength objectives and a negative impact on the thermal objective.

The definition of latency is used to obtain constraint (23). If there is no flip-flop (FF) on a wire (i, j), the delay of this wire is calculated as $d(i, j) = d_r(X_{ij} + Y_{ij})$. Then, $g_i + d(i, j)$

represents the latency of module i accessing module j, where $d(i, j)$ denotes the delay between i and j. Since C denotes the clock period constraint, $(g_i + d(i, j))/C$ denotes the minimum number of FFs required on (i, j) to satisfy C. Absolute values on x and y distance are given in (24)–(25). Constraint (26) requires that the number of FFs on each edge is non-negative. The block boundary constraints (30)–(33) require that all modules in the block be enclosed by these block boundaries. The center of gravity constraints (34)–(35) require that the module area-weighted mean (center of gravity) among all modules in each sub-block corresponds to the center of the sub-block.

REFERENCES

[1] K. Skadron, M. R. Stan, K. Sankaranarayanan, W. Huang, S. Velusamy, and D. Tarjan, "Temperature-aware microarchitecture: Modeling and implementation," *ACM Transactions on Architecture and Code Optimization*, vol. 1, no. 1, pp. 94–125, 2004.

[2] C. Blat, E. Nicollian, and E. Poindexter, "Mechanism of negative-bias-temperature instability," *Journal of Applied Physics*, vol. 69, pp. 1712–1720, 1991.

[3] N. Sa, J. Kang, H. Yang, X. Liu, Y. He, R. Han, C. Ren, H. Yu, D. Chan, and D.-L. Kwong, "Mechanism of positive-bias temperature instability in sub-1-nm TaN/HfN/HfO2 gate stack with low preexisting traps," *IEEE Electron Device Letters*, vol. 26, no. 9, pp. 610–612, 2005.

[4] L. Stockmeyer, "Optimal orientations of cells in slicing floorplan designs," *Information and Control*, vol. 57, no. 2-3, pp. 91–101, 1983.

[5] D. Wong and C. Liu, "A new algorithm for floorplan design," in *Proceedings IEEE/ACM Design Automation Conference*, pp. 101–107, 1986.

[6] D. Wong and C. Liu, "Floorplan design for rectangular and L-shaped modules," in *Proceedings IEEE/ACM International Conference on Computer-Aided Design*, pp. 101–107, 1987.

[7] S. Sutanthavibul, E. Shragowitz, and J. Rosen, "An analytical approach to floorplan design and optimization," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 761–769, 1991.

[8] H. Murata, K. Fujiyoshi, and M. Kaneko, "VLSI/PCB placement with obstacles based on sequence pair," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 60–68, 1998.

[9] S. Nakatake, M. Furuya, and Y. Kajitani, "Module placement on BSG-structure with pre-placed modules and rectilinear modules," in *Proceedings Asia and South Pacific Design Automation Conference*, pp. 571–576, feb 1998.

[10] M. Rebaudengo and M. Reorda, "GALLO: A genetic algorithm for floorplan area optimization," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 15, pp. 943–951, aug 1996.

[11] J. Cong, A. Jagannathan, G. Reinman, and M. Romesis, "Microarchitecture evaluation with physical planning," in *Proceedings IEEE/ACM Design Automation Conference*, pp. 634–639, 2003.

[12] M. Casu and L. Macchiarulo, "Floorplanning for throughput," in *Proceedings International Symposium on Physical Design*, pp. 62–69, 2004.

[13] M. Ekpanyapong, J. Minz, T. Watewai, H.-H. Lee, and S. K. Lim, "Profile-guided microarchitectural floorplanning for deep submicron processor design," in *Proceedings IEEE/ACM Design Automation Conference*, pp. 634–639, 2004.

[14] C. Long, L. Simonson, W. Liao, and L. He, "Floorplanning optimization with trajectory piecewise-linear model for pipelined interconnects," in *Proceedings IEEE/ACM Design Automation Conference*, pp. 640–645, 2004.

[15] V. Nookala, Y. Chen, D. J. Lilja, and S. S. Sapatnekar, "Microarchitecture-aware floorplanning using a statistical design of experiments approach," in *Proceedings IEEE/ACM Design Automation Conference*, pp. 579–584, 2005.

[16] D. C. Montgomery, *Design and analysis of experiments*. New York, NY: John Wiley, 1991.

[17] L. Mercado and V. Sarihan, "Evaluation of die edge cracking in flip-chip PBGA packages," *IEEE Transactions on Components and Packaging Technologies*, vol. 26, pp. 719–723, dec. 2003.

[18] J. Choi, C.-Y. Cher, H. Franke, H. Hamann, A. Weger, and P. Bose, "Thermal-aware task scheduling at the system software level," in *Proceedings International Symposium on Low Power Electronics and Design*, (New York, NY, USA), pp. 213–218, ACM, 2007.

[19] C. Tsai and S. Kang, "Cell-level placement for improving substrate thermal distribution," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 253–266, 2000.

[20] C. N. Chu and D. F. Wong, "A matrix synthesis approach to thermal placement," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 163–168, 1998.

[21] B. Obermeier and F. Johannes, "Temperature-aware global placement," in *Proceedings Asia and South Pacific Design Automation Conference*, pp. 143–148, 2004.

[22] G. Chen and S. Sapatnekar, "Partition-driven standard cell thermal placement," in *Proceedings International Symposium on Physical Design*, pp. 75–80, 2003.

[23] K. Balakrishnan, V. Nanda, S. Easwar, and S. K. Lim, "Wire congestion and thermal aware 3D global placement," in *Proceedings Asia and South Pacific Design Automation Conference*, pp. 1131–1134, 2005.

[24] J. Cong, J. Wei, and Y. Zhang, "A thermal-driven floorplanning algorithm for 3D ICs," in *Proceedings IEEE/ACM International Conference on Computer-Aided Design*, pp. 306–313, 2004.

[25] J.-M. Lin and Y.-W. Chang, "TCG: A transitive closure graph-based representation for general floorplans," *IEEE Transactions on VLSI Systems*, vol. 13, pp. 288–292, feb. 2005.

[26] W. Hung, Y. Xie, N. Vijaykrishnan, C. Addo-Quaye, T. Theocharides, and M. Irwin, "Thermal-aware floorplanning using genetic algorithms," in *Proceedings International Symposium on Quality Electronic Design*, pp. 634–639, 2005.

[27] M. Ekpanyapong, *Microarchitecture-Aware Physical Planning for Deep Submicron Technology*. Doctoral Thesis, Georgia Institute of Technology, 2006.

[28] K. Skadron, M. Stan, W. Huang, S. Velusamy, K. Sankaranarayanan, and D. Tarjan, "Temperature-aware microarchitecture," in *Proceedings IEEE International Symposium on Computer Architecture*, pp. 2–13, 2003.

[29] M. Huang, J. Renau, S.-M. Yoo, and J. Torrellas, "A framework for dynamic energy efficiency and temperature management," in *Proceedings Annual IEEE/ACM International Symposium on Microarchitecture*, (Monterey, California), pp. 202–213, 2000.

[30] D. Brooks and M. Martonosi, "Dynamic thermal management for high-performance microprocessors," in *Proceedings IEEE International Symposium on High-Performance Computer Architecture*, (Monterrey, Mexico), pp. 171–182, IEEE Computer Society, 2001.

[31] S. Dropsho, V. Kursun, D. Albonesi, S. Dwarkadas, and E. Friedman, "Managing static leakage energy in microprocessor functional units," in *Proceedings Annual IEEE/ACM International Symposium on Microarchitecture*, pp. 321–332, 2004.

[32] N. Kim, K. Flautner, D. Blaauw, and T. Mudge, "Drowsy instruction caches: Leakage power reduction using dynamic voltage scaling and cache sub-bank prediction," in *Proceedings Annual IEEE/ACM International Symposium on Microarchitecture*, pp. 219–230, 2002.

[33] D. Duarte, Y. Tsai, N. Vijaykrishnan, and M. Irwin, "Evaluating run-time techniques for leakage power reduction," in *Proceedings Asia and South Pacific Design Automation Conference*, pp. 31–38, 2002.

[34] S. Kaxiras, Z. Hu, and M. Martonosi, "Cache decay: Exploiting generational behavior to reduce cache leakage power," in *Proceedings IEEE International Symposium on Computer Architecture*, pp. 240–251, 2001.

[35] L. He, W. Liao, and M. Stan, "System level leakage reduction considering leakage and thermal interdependency," in *Proceedings IEEE/ACM Design Automation Conference*, pp. 12–17, 2004.

[36] F. Li, C. Nicopoulos, T. Richardson, Y. Xie, V. Narayanan, and M. Kandemir, "Design and management of 3D chip multiprocessors using network-in-memory," in

Proceedings IEEE International Symposium on Computer Architecture, pp. 130–141, 2006.

[37] P. Chaparro, J. Gonzalez, G. Magklis, Q. Cai, and A. Gonzalez, "Understanding the thermal implications of multi-core architectures," *IEEE Transactions on Parallel and Distributed Systems*, vol. 18, pp. 1055–1065, 2007.

[38] U. Y. Ogras and R. Marculescu, "Energy- and performance- driven NoC communication architecture synthesis using a decomposition approach," in *Proceedings Design, Automation and Test in Europe*, pp. 352–357, 2005.

[39] Q. K. Zhou, *Power Distribution Network Design for VLSI*. Wiley-IEEE, 2004.

[40] K. Wong, T. Rahal-Arabi, M. Ma, and G. Taylor, "Enhancing microprocessor immunity to power supply noise with clock-data compensation," *IEEE Journal of Solid-State Circuits*, vol. 41, pp. 749–758, april 2006.

[41] G. Huang, A. Naeemi, T. Zhou, D. O'Connor, A. Muszynski, B. Singh, D. Becker, J. Venuto, and J. Meindl, "Compact physical models for chip and package power and ground distribution networks for gigascale integration (GSI)," in *Proceedings IEEE Electronic Components and Technology Conference*, pp. 646–651, june 2008.

[42] E. Grochowski, D. Ayers, and V. Tiwari, "Microarchitectural simulation and control of di/dt-induced power supply voltage variation," in *Proceedings IEEE International Symposium on High-Performance Computer Architecture*, pp. 7–16, 2002.

[43] S. Zhao, C. Koh, and K. Roy, "Decoupling capacitance allocation and its application to power supply noise aware floorplanning," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 81–92, 2002.

[44] H. Chen, L. Huang, I. Liu, M. Lai, and D. Wong, "Floorplanning with power supply noise avoidance," in *Proceedings Asia and South Pacific Design Automation Conference*, pp. 427–430, 2003.

[45] Y. Chen, K. Roy, and C.-K. Koh, "Current demand balancing: A technique for minimization of current surge in high performance clock-gated microprocessors," *IEEE Transactions on VLSI Systems*, pp. 75–85, 2005.

[46] J. Minz, S. K. Lim, and C. K. Koh, "3D module placement for congestion and power noise reduction," in *Proceedings Great Lakes Symposum on VLSI*, pp. 458–461, 2005.

[47] R. Joseph, D. Brooks, and M. Martonosi, "Control techniques to eliminate voltage emergencies in high performance processors," in *Proceedings IEEE International Symposium on High-Performance Computer Architecture*, pp. 79–90, 2003.

[48] R. Joseph, Z. Hu, and M. Martonosi, "Wavelet analysis for microprocessor design: Experiences with wavelet-based di/dt characterization," in *Proceedings IEEE International Symposium on High-Performance Computer Architecture*, pp. 36–46, 2004.

[49] M. D. Powell and T. N. Vijaykumar, "Exploiting resonant behavior to reduce inductive noise," in *Proceedings IEEE International Symposium on Computer Architecture*, pp. 288–299, 2004.

[50] M. D. Powell and T. N. Vijaykumar, "Pipeline damping: A microarchitectural technique to reduce inductive noise in supply voltage," in *Proceedings IEEE International Symposium on Computer Architecture*, pp. 72–83, 2003.

[51] K. Hazelwood and D. Brooks, "Eliminating voltage emergencies via microarchitectural voltage control feedback and dynamic optimization," in *Proceedings International Symposium on Low Power Electronics and Design*, pp. 326–331, 2004.

[52] M. D. Powell and T. N. Vijaykumar, "Pipeline muffling and a priori current ramping: Architectural techniques to reduce high-frequency inductive noise," in *Proceedings International Symposium on Low Power Electronics and Design*, pp. 223–228, 2003.

[53] Z. Tang, N. Chang, S. Lin, W. Xie, S. Nakagawa, and L. He, "Ramp up/down floating point unit to reduce inductive noise," in *Workshop on Power-Aware Computer Systems*, 2000.

[54] S. Das, A. Chandrakasan, and R. Reif, "Design tools for 3-D integrated circuits," in *Proceedings Asia and South Pacific Design Automation Conference*, pp. 53–56, 2003.

[55] B. Goplen and S. Sapatnekar, "Efficient thermal placement of standard cells in 3D ICs using a force directed approach," in *Proceedings IEEE/ACM International Conference on Computer-Aided Design*, pp. 86–89, 2003.

[56] T. Tanprasert, "An analytical 3-D placement that reserves routing space," in *Proceedings IEEE International Symposium on Circuits and Systems*, pp. 69–72, 2000.

[57] I. Kaya, M. Olbrich, and E. Barke, "3-D placement considering vertical interconnects," in *Proceedings IEEE International SOC Conference*, pp. 257–258, 2003.

[58] D. H. Kim, K. Athikulwongse, and S. K. Lim, "A study of through-silicon-via impact on the 3D stacked IC layout," in *Proceedings IEEE/ACM International Conference on Computer-Aided Design*, pp. 674–680, 2009.

[59] J. Joyner, P. Zarkesh-Ha, J. Davis, and J. Meindl, "A three-dimensional stochastic wire-length distribution for variable separation of strata," in *Proceedings IEEE International Interconnect Technology Conference*, pp. 126–128, 2000.

[60] R. Zhang, K. Roy, C.-K. Koh, and D. B. Janes, "Exploring SOI device structures and interconnect architectures for 3-dimensional integration," in *Proceedings IEEE/ACM Design Automation Conference*, pp. 846–851, 2001.

[61] D. H. Kim, S. Mukhopadhyay, and S. K. Lim, "TSV-aware interconnect length and power prediction for 3D stacked ICs," in *Proceedings IEEE International Interconnect Technology Conference*, pp. 26–28, 2009.

[62] L. Cheng, W. Hung, G. Yang, and X. Song, "Congestion estimation for 3-D circuit architectures," *IEEE Transactions On Circuits and Systems II: Express Briefs*, pp. 655–659, 2004.

[63] J. Cong and Y. Zhang, "Thermal-driven multilevel routing for 3-D ICs," in *Proceedings Asia and South Pacific Design Automation Conference*, pp. 121–126, 2005.

[64] B. Goplen and S. Sapatnekar, "Thermal via placement in 3-D ICs," in *Proceedings International Symposium on Physical Design*, pp. 167–174, 2005.

[65] V. Pavlidis and E. Friedman, "Interconnect delay minimization through interlayer via placement in 3-D ICs," in *Proceedings Great Lakes Symposum on VLSI*, pp. 20–25, 2005.

[66] Y. Deng and W. Maly, "Physical design of the 2.5D stacked system," in *Proceedings IEEE International Conference on Computer Design*, pp. 211–214, 2003.

[67] P. Shiu, R. Ravichandran, S. Easwar, and S. Lim, "Multi-layer floorplanning for reliable system-on-package," in *Proceedings IEEE International Symposium on Circuits and Systems*, pp. 69–72, may 2004.

[68] L. Cheng, L. Deng, and M. Wong, "Floorplanning for 3-D VLSI design," in *Proceedings Asia and South Pacific Design Automation Conference*, pp. 405–411, 2005.

[69] G. Huang, M. Bakir, A. Naeemi, H. Chen, and J. Meindl, "Power delivery for 3D chip stacks: Physical modeling and design implication," in *Proceedings IEEE Conference on Electrical Performance of Electronic Packaging*, pp. 205–208, 2007.

[70] P. Jain, T.-H. Kim, J. Keane, and C. H. Kim, "A multi-story power delivery technique for 3D integrated circuits," in *Proceedings International Symposium on Low Power Electronics and Design*, (New York, NY, USA), pp. 57–62, ACM, 2008.

[71] J. Gu and C. H. Kim, "Multi-story power delivery for supply noise reduction and low voltage operation," in *Proceedings International Symposium on Low Power Electronics and Design*, (New York, NY, USA), pp. 192–197, ACM, 2005.

[72] H. Yu, J. Ho, and L. He, "Allocating power ground vias in 3D ICs for simultaneous power and thermal integrity," *ACM Transactions on Design Automation of Electronics Systems*, vol. 14, no. 3, pp. 1–31, 2009.

[73] T. Thorolfsson, K. Gonsalves, and P. Franzon, "Design automation for a 3DIC FFT processor for synthetic aperture radar: A case study," in *Proceedings IEEE/ACM Design Automation Conference*, pp. 51–56, 2009.

[74] P. Franzon, W. Davis, M. Steer, S. Lipa, E. C. Oh, T. Thorolfsson, T. Doxsee, S. Berkeley, B. Shani, and K. Obermiller, "Design and CAD for 3D integrated circuits," in *Proceedings IEEE/ACM Design Automation Conference*, pp. 668–673, 2008.

[75] Massachusetts Institute of Technology Lincoln Labs, "*MITLL Low-Power FDSOI CMOS Process Design Guide*." revision 2008:6 edition. September 2008.

[76] P. Shivakumar and N. P. Jouppi, "CACTI 3.0: An integrated cache timing, power, and area model," Tech. Rep. 2001.2, HP Western Research Labs, 2001.

[77] J. C. Eble, V. K. De, D. S. Wills, and J. D. Meindl, "A generic system simulator (GENESYS) for ASIC technology and architecture beyond 2001,"

[78] T. M. Austin, "Simplescalar tool suite." http:/www.simplescalar.com, January 2006.

[79] D. Brooks, V. Tiwari, and M. Martonosi, "Wattch: A framework for architectural-level power analysis and optimizations," in *Proceedings IEEE International Symposium on Computer Architecture*, pp. 83–94, 2000.

[80] D. Duarte, Vijaykrishnan, and M. J. Erwin, "A clock power model to evaluate the impact of architectural and technology optimizations," *IEEE Transactions on VLSI Systems*, vol. 10, pp. 844–855, 2002.

[81] Y. Tsai, A. Ankadi, N. Vijaykrishnan, M. Irwin, and T. Theocharides, "ChipPower: An architecture-level leakage simulator," in *Proceedings IEEE International SOC Conference*, pp. 395–398, 2004.

[82] eCACTI, January 2005. http://www.ics.uci.edu/ maheshmn/eCACTI/main.htm.

[83] W. Liao, F. Li, and L. He, "Microarchitecture level power and thermal simulation considering temperature," in *Proceedings International Symposium on Low Power Electronics and Design*, pp. 211–216, 2003.

[84] H. Murata, K. Fujiyoshi, S. Nakatake, and Y. Kajitani, "Rectangle packing based module placement," in *Proceedings IEEE/ACM International Conference on Computer-Aided Design*, pp. 472–479, 1995.

[85] S. B. Horn, "Vertically integrated sensor arrays: VISA," in *Defense and Security Symposium*, pp. 332–340, 2004.

[86] J. Cong and S. K. Lim, "Edge separability based circuit clustering with application to multi-level circuit partitioning," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 23, no. 3, pp. 346–357, 2004.

[87] J. C. Chi and M. C. Chi, "An effective soft module floorplanning algorithm based on sequence pair," in *Proceedings IEEE International ASIC/SOC Conference*, pp. 54–58, 2002.

[88] D. H. Kim and S. K. Lim, "Bus-aware microarchitectural floorplanning," in *Proceedings Asia and South Pacific Design Automation Conference*, pp. 204–208, 2008.

[89] S. C. Woo, M. Ohara, E. Torrie, J. P. Singh, and A. Gupta, "The SPLASH-2 programs: Characterization and methodological considerations," in *Proceedings IEEE International Symposium on Computer Architecture*, pp. 24–36, 1995.

[90] J. Renau, B. Fraguela, J. Tuck, W. Liu, M. Prvulovic, L. Ceze, S. Sarangi, P. Sack, K. Strauss, and P. Montesinos, "SESC simulator," January 2005. http://sesc.sourceforge.net.

[91] H. Jacobson, P. Bose, Z. Hu, A. Buyuktosunoglu, V. Zyuban, R. Eickemeyer, L. Eisen, J. Griswell, D. Logan, B. Sinharoy, and J. Tendler, "Stretching the limits of clock-gating efficiency in server-class processors," in *Proceedings IEEE International Symposium on High-Performance Computer Architecture*, pp. 238–242, 2005.

[92] M. K. Gowan, L. L. Biro, and D. B. Jackson, "Power considerations in the design of the alpha 21264 microprocessor," in *Proceedings IEEE/ACM Design Automation Conference*, pp. 726–731, 1998.

[93] M. D. Pant, P. Pant, D. S. Wills, and V. Tiwari, "Inductive noise reduction at the architectural level," in *Proceedings International Conference on VLSI Design*, pp. 162–167, 2000.

[94] F. Mohamood, M. B. Healy, S. K. Lim, and H.-H. S. Lee, "A floorplan-aware dynamic inductive noise controller for reliable processor design," in *Proceedings Annual IEEE/ACM International Symposium on Microarchitecture*, pp. 3–14, 2006.

[95] F. Mohamood, M. B. Healy, S. K. Lim, and H.-H. S. Lee, "Noise-Direct: A technique for power supply noise aware floorplanning using microarchitecture profiling," in *Proceedings Asia and South Pacific Design Automation Conference*, pp. 786–791, 2007.

[96] M. B. Healy, F. Mohamood, H.-H. S. Lee, and S. K. Lim, "A unified methodology for power supply noise reduction in modern microarchitecture design," in *Proceedings Asia and South Pacific Design Automation Conference*, pp. 611–616, 2008.

[97] E. Wong, J. Minz, and S. K. Lim, "Decoupling capacitor planning and sizing for noise and leakage reduction," in *Proceedings IEEE/ACM International Conference on Computer-Aided Design*, pp. 395–400, 2006.

[98] T. M. Austin and G. S. Sohi, "Zero-cycle loads: Microarchitecture support for reducing load latency," in *Proceedings Annual IEEE/ACM International Symposium on Microarchitecture*, pp. 82–92, 1995.

[99] M. Healy, M. Vittes, M. Ekpanyapong, C. Ballapuram, S. K. Lim, H.-H. S. Lee, and G. H. Loh, "Microarchitectural floorplanning under performance and temperature tradeoff," in *Proceedings of the Design, Automation and Test in Europe*, pp. 1288–1293, 2006.

[100] H. Eisenmann and F. M. Johannes, "Generic global placement and floorplanning," in *Proceedings IEEE/ACM Design Automation Conference*, pp. 269–274, 1998.

[101] D. Sekar, C. King, B. Dang, T. Spencer, H. Thacker, P. Joseph, M. Bakir, and J. Meindl, "A 3D-IC technology with integrated microchannel cooling," in *Proceedings IEEE International Interconnect Technology Conference*, pp. 13–15, 2008.

[102] M. Healy, M. Vittes, M. Ekpanyapong, C. Ballapuram, S. K. Lim, H.-H. S. Lee, and G. H. Loh, "Multi-objective microarchitectural floorplanning for 2D and 3D ICs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1288–1293, 2006.

[103] G. H. Loh, "3D-stacked memory architectures for multi-core processors," in *Proceedings IEEE International Symposium on Computer Architecture*, pp. 453–464, 2008.

[104] B. Dang, S. Wright, P. Andry, C. Tsang, C. Patel, R. Polastre, R. Horton, K. Sakuma, B. Webb, E. Sprogis, G. Zhang, A. Sharma, and J. Knickerbocker, "Assembly, characterization, and reworkability of Pb-free ultra-fine pitch C4s for system-on-package," in *Proceedings IEEE Electronic Components and Technology Conference*, pp. 42–48, may 2007.

[105] P. Leduc, M. Assous, L. Di Cioccio, M. Zussy, T. Signamarcheix, A. Roman, M. Rousseau, S. Verrun, L. Bally, D. Bouchu, L. Cadix, A. Farcy, and N. Sillon, "First integration of Cu TSV using die-to-wafer direct bonding and planarization," in *Proceedings IEEE Conference on 3D System Integration*, pp. 1–5, 28-30 2009.

[106] Synopsys, "Raphael." http://www.synopsys.com, May 2008.

[107] F. Grover, *Inductance Calculations: Working Formulas and Tables*. Dover Publications.

[108] V. George, S. Jahagirdar, C. Tong, K. Smits, S. Damaraju, S. Siers, V. Naydenov, T. Khondker, S. Sarkar, and P. Singh, "Penryn: 45-nm next generation intel coreTM 2 processor," in *Proceedings IEEE Asian Solid-State Circuits Conference*, pp. 14–17, 2007.

[109] K. Puttaswamy and G. H. Loh, "Thermal herding: Microarchitecture techniques for controlling hotspots in high-performance 3D-integrated processors," in *Proceedings IEEE International Symposium on High-Performance Computer Architecture*, pp. 193–204, 2007.

[110] J. Lee, H. Kim, K. Kyung, M. You, H. Lee, K. Park, and B. Chung, "Design, analysis, and optimization of DDR2 memory power delivery network," in *Proceedings IEEE Conference on Electrical Performance of Electronic Packaging*, pp. 87–90, 2007.

[111] C.-W. Ho, A. E. Ruehli, and P. A. Brennan, "The modified nodal approach to network analysis," *IEEE Transactions on Circuits and Systems*, pp. 504–509, June 1975.

[112] Q. Zhou, K. Sun, K. Mohanram, and D. C. Sorensen, "Large power grid analysis using domain decomposition," in *Proceedings Design, Automation and Test in Europe*, pp. 27–32, 2006.

[113] J.-M. Koo, S. Im, L. Jiang, and K. E. Goodson, "Integrated microchannel cooling for three-dimensional electronic architecture," *J. Heat Transfer*, pp. 49–58, 2005.

[114] S. V. Patankar, *Numerical Heat Transfer and Fluid Flow*. Hemisphere, 1980.

[115] M. O. McLinden, S. Klein, E. Lemmon, and A. Peskin, "NIST thermodynamic and transport properties of refrigerants and refrigerant mixtures database (REFPROP)." Ver. 6.0, National Institute of Standards and Technology, Gaithersburg, Maryland, USA, 1998.

[116] M. B. Healy and S. K. Lim, "A study of stacking limit and scaling in 3D ICs: An interconnect perspective," in *Proceedings IEEE Electronic Components and Technology Conference*, pp. 1213–1220, May 2009.

[117] J. Burns, B. Aull, C. Chen, C.-L. Chen, C. Keast, J. Knecht, V. Suntharalingam, K. Warner, P. Wyatt, and D. Yost, "A wafer-scale 3-D circuit integration technology," *IEEE Transactions on Electron Devices*, vol. 53, pp. 2507–2516, October 2006.

[118] M. Koyanagi, T. Fukushima, and T. Tanaka, "Three-dimensional integration technology and integrated systems," in *Proceedings Asia and South Pacific Design Automation Conference*, pp. 409–415, 2009.

[119] X. Dong and Y. Xie, "System-level cost analysis and design exploration for 3D ICs," in *Proceedings Asia and South Pacific Design Automation Conference*, pp. 234–241, 2009.

[120] M. B. Healy, K. Athikulwongse, R. Goel, M. M. Hossain, D. H. Kim, Y.-J. Lee, D. L. Lewis, T.-W. Lin, C. Liu, M. Jung, B. Ouellette, M. Pathak, H. Sane, G. Shen, D. H. Woo, X. Zhao, G. H. Loh, H.-H. S. Lee, and S. K. Lim, "Design and analysis of 3D-MAPS: A many-core 3D processor with stacked memory," in *Proceedings IEEE Custom Integrated Circuits Conference*, 2010.

[121] Y. Deng and W. Maly, "2.5-dimensional VLSI system integration," *IEEE Transactions on VLSI Systems*, vol. 13, no. 6, pp. 668–677, 2005.

R.1 Related Publications

This dissertation is based on and/or related to the work and results presented in the following publications in print:

[1] **M. B. Healy**, M. Vittes, M. Ekpanyapong, C. Ballapuram, S. K. Lim, H-H. S. Lee, and G. H. Loh, "Microarchitectural floorplanning under performance and temperature tradeoff," in *Proceedings Design, Automation and Test in Europe*, pp. 1–6, March 2006.

[2] F. Mohamood, **M. B. Healy**, S. K. Lim, and H.-H. S. Lee, "A floorplan-aware dynamic inductive noise controller for reliable processor design," in *Proceedings Annual IEEE/ACM International Symposium on Microarchitecture*, pp. 3–14, December 2006.

[3] **M. B. Healy**, M. Vittes, M. Ekpanyapong, S. K. Lim, H-H. S. Lee, and G. H. Loh, "Multi-objective microarchitectural floorplanning for 2D and 3D ICs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 38–52, January 2007.

[4] F. Mohamood, **M. B. Healy**, S. K. Lim, and H.-H. S. Lee, "Noise-Direct: A technique for power supply noise aware floorplanning using microarchitecture profiling," in *Proceedings Asia and South Pacific Design Automation Conference*, pp. 786–791, January 2007.

[5] **M. B. Healy**, F. Mohamood, H.-H. S. Lee, and S. K. Lim, "A unified methodology for power supply noise reduction in modern microarchitecture design," in *Proceedings Asia and South Pacific Design Automation Conference*, pp. 611–616, January 2008.

[6] **M. B. Healy**, H.-H. S. Lee, G. H. Loh, and S. K. Lim, "Thermal optimization in multi-granularity multi-core floorplanning," in *Proceedings Asia and South Pacific Design Automation Conference*, pp. 43–48, January 2009.

[7] **M. B. Healy** and S. K. Lim, "A study of stacking limit and scaling in 3D ICs: An interconnect perspective," in *Proceedings IEEE Electronic Components and Technology Conference*, pp. 1213–1220, May 2009.

[8] Y.-J. Lee, **M. B. Healy**, and Sung Kyu Lim, "Co-design of reliable signal and power interconnects in 3D stacked ICs," in *Proceedings IEEE International Interconnect Technology Conference*, pp. 56–58, June 2009.

[9] **M. B. Healy** and S. K. Lim, "Power delivery system architecture for many-tier 3D systems," in *Proceedings IEEE Electronic Components and Technology Conference*, pp. 1682–1688, June 2010.

In addition, this dissertation is based on and/or related to the work and results presented in the following papers accepted/in submission:

[10] **M. B. Healy**, K. Athikulwongse, R. Goel, M. M. Hossain, D. H. Kim, Y.-J. Lee, D. L. Lewis, T.-W. Lin, C. Liu, M. Jung, B. Ouellette, M. Pathak, H. Sane, G. Shen, D. H. Woo, X. Zhao, G. H. Loh, H.-H. S. Lee, and S. K. Lim, "Design and analysis of 3D-MAPS: A many-core 3D processor with stacked memory," accepted by *IEEE Custom Integrated Circuits Conference*, September 2010.

[11] **M. B. Healy**, F. Mohamood, H.-H. S. Lee, and S. K. Lim, "Integrated microarchitectural floorplanning and runtime controller for inductive noise mitigation," submitted to *ACM Transactions on Design Automation of Electronic Systems*, 2010.

[12] **M. B. Healy** and S. K. Lim, "Temperature and power-supply-noise scaling in many-tier 3D systems," to be submitted to *ACM Journal on Emerging Technologies in Computing Systems*, 2010.

[13] **M. B. Healy** and S. K. Lim, "Novel through-silicon-via topology for power-supply-noise reduction," to be submitted to *IEEE Transactions on VLSI Systems*, 2010.

The author has also completed work unrelated to this dissertation presented in the following publications in print:

[14] **M. B. Healy**, S. Ravindran, and D. Anderson, "Effects of varying parameters in asymmetric adaboost on the accuracy of a cascade audio classifier," in *IEEE Southeast Regional Conference*, pp. 169–172, March 2004.

[15] M. Ekpanyapong, **M. B. Healy**, and S. K. Lim, "Placement for configurable dataflow architecture," in *Proceedings Asia South Pacific Design Automation Conference*, pp. 1127–1130, January 2005.

[16] **M. B. Healy**, M. Ekpanyapong, and S. K. Lim, "Placement and routing for configurable dataflow architecture," in *Proceedings International Conference on Field Programmable Logic and Applications*, pp. 71–76, August 2005.

[17] M. Ekpanyapong, **M. B. Healy**, and S. K. Lim, "Profile-driven instruction mapping for dataflow architectures," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 3017–3025, December 2006.